

6609091

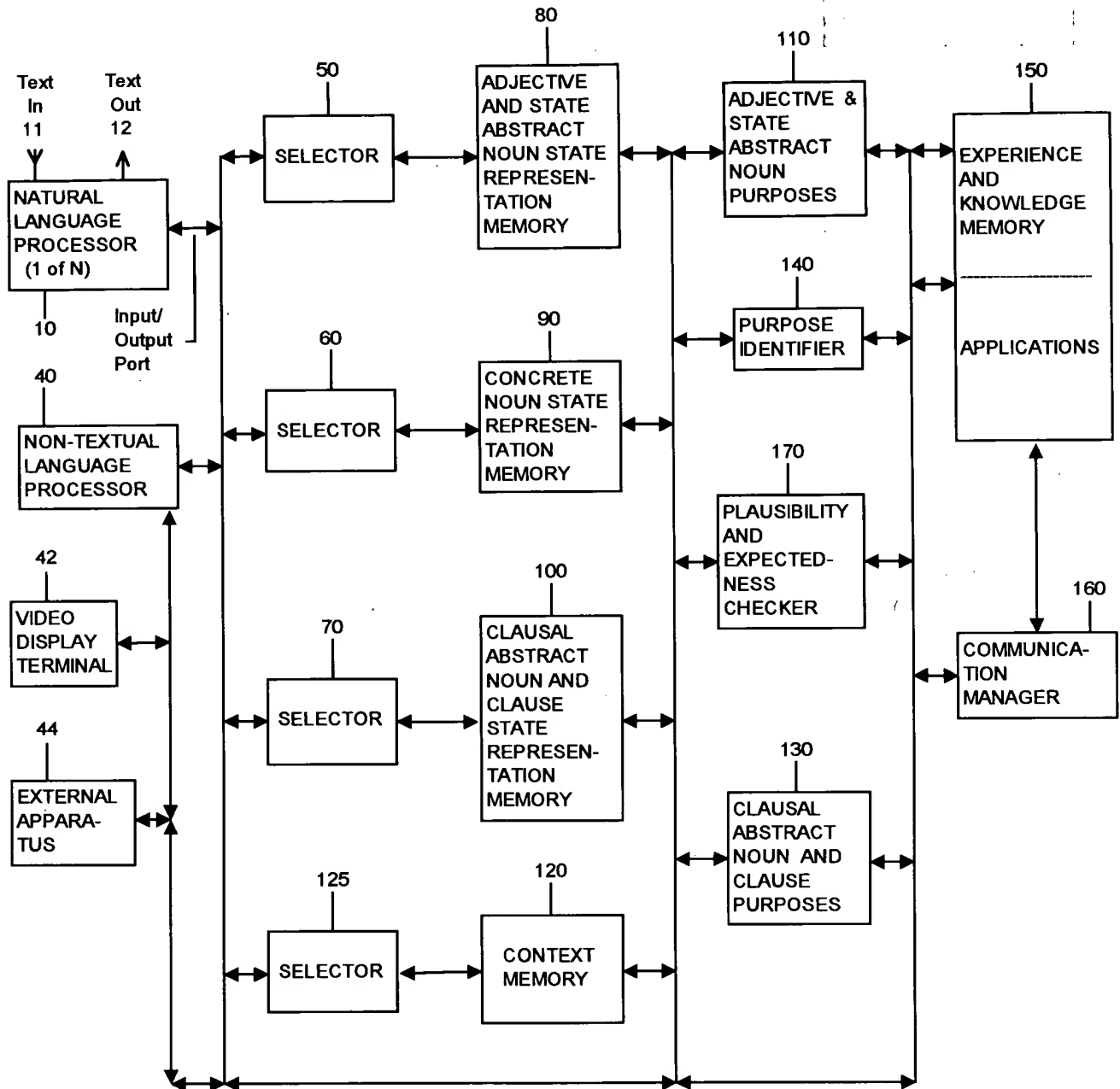


FIG. 1

002260" 085T 2960

```
graph TD
    12[WORD ISOLATION STEP 12] <--> 14[DICTIONARY LOOK UP STEP 14]
    14 <--> 16[SYNTAX PARSE STEP 16]
    16 <--> 18[DICTIONARY LOOKUP STEP 18]
    18 <--> 20[DICTIONARY 20]
    20 <--> 22[FUNCTION WORD PROCESSING STEP 22]
    22 <--> 24[MORPHOLOGICAL PROCESSING STEP 24]
    24 <--> 26[ELLIPSIS PROCESSING STEP 26]
    26 <--> 12
    200[TEXT GENERATION STEP 200] <--> 30[SYNTAX PARSE TREES 30]
    30 <--> 20
    30 <--> 28[DATA STORAGE]
    28 <--> 22
```

The flowchart illustrates a natural language processing system. It consists of two main vertical paths of steps. The left path includes: WORD ISOLATION STEP 12, DICTIONARY LOOK UP STEP 14, SYNTAX PARSE STEP 16, DICTIONARY LOOKUP STEP 18, FUNCTION WORD PROCESSING STEP 22, MORPHOLOGICAL PROCESSING STEP 24, and ELLIPSIS PROCESSING STEP 26. The right path includes: TEXT GENERATION STEP 200, SYNTAX PARSE TREES 30, DICTIONARY 20, and DATA STORAGE. Bidirectional arrows connect the steps in each path, indicating a two-way flow of information. Specifically, steps 12-14, 14-16, 16-18, 18-20, 20-22, 22-24, and 24-26 are connected on the left. On the right, steps 200-30, 30-20, and 30-28 are connected. Additionally, a bidirectional arrow connects step 28 to step 22.

FIG. 2

TEXT OF WORD	REPRESENTATION NUMBER	SYNTAX WORDSETS	STATE/FUNCTION ADDRESSES	ANOMALIES	POINTERS TO COMMON TABLES
Text word	Number used to the represent the text word	Set of wordsets associated with the text word; Each wordset has its part of speech and a pointer to a set of pointers to phrases started with the text word with or without ellipsis stored in a common table	Associated with each wordset is a set of word sense numbers or an address to the function word process with an associated function code	Partitioned by wordset; each entry has: type of anomaly, criteria for selecting the anomaly, an address of a replacement process and/or a replacement data structure	Addresses of common tables associated with a wordset

FIG. 3A

ENTRY NUMBER	STATE REPRESENTATION ADDRESS	SET OF BASE WORD ENTRIES
Word sense number	Address of data structure of the word sense number	Dictionary 20 base word entries implying the word sense number

FIG. 3B

002260" 08572960

TEXT OF AFFIX OR INFLECTION	REPRESENTATION NUMBER
Text	Affix code or inflection code

FIG. 3C

002250-092700

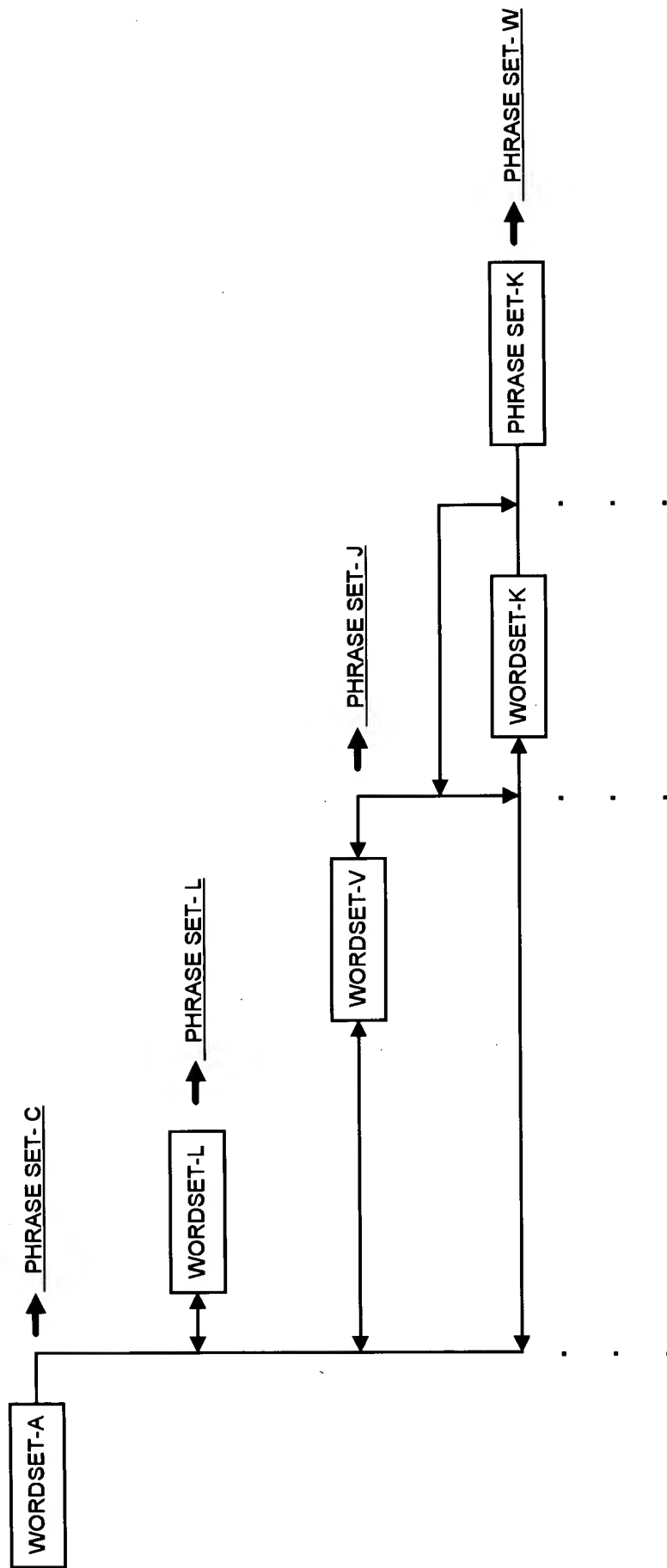


FIG. 4A

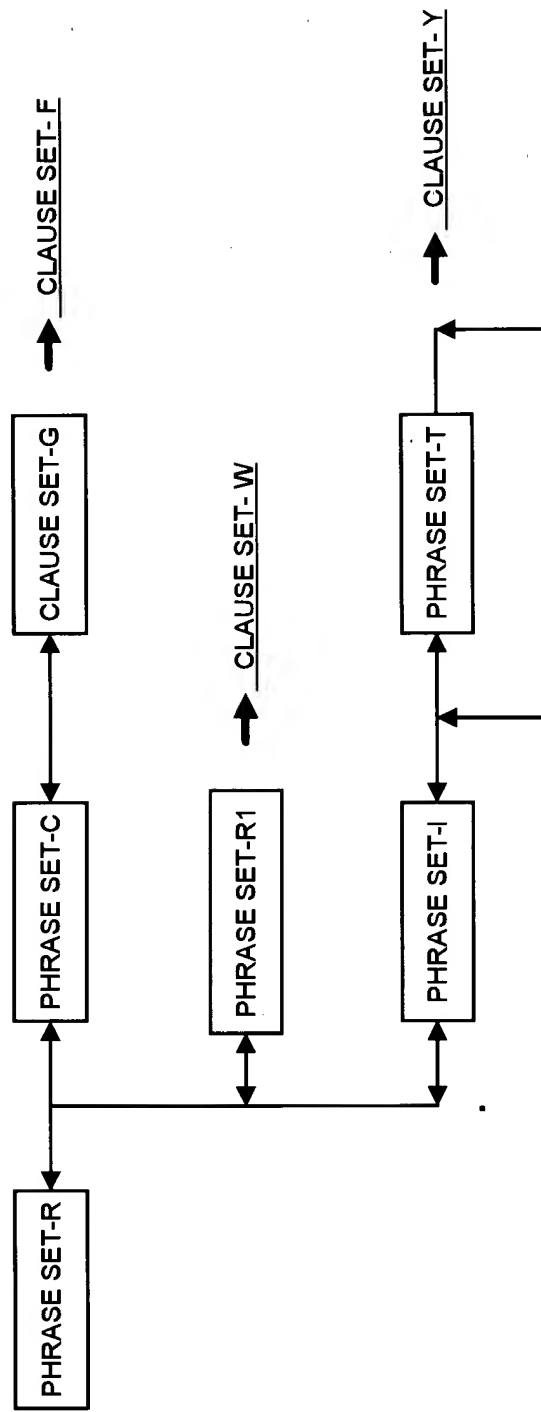
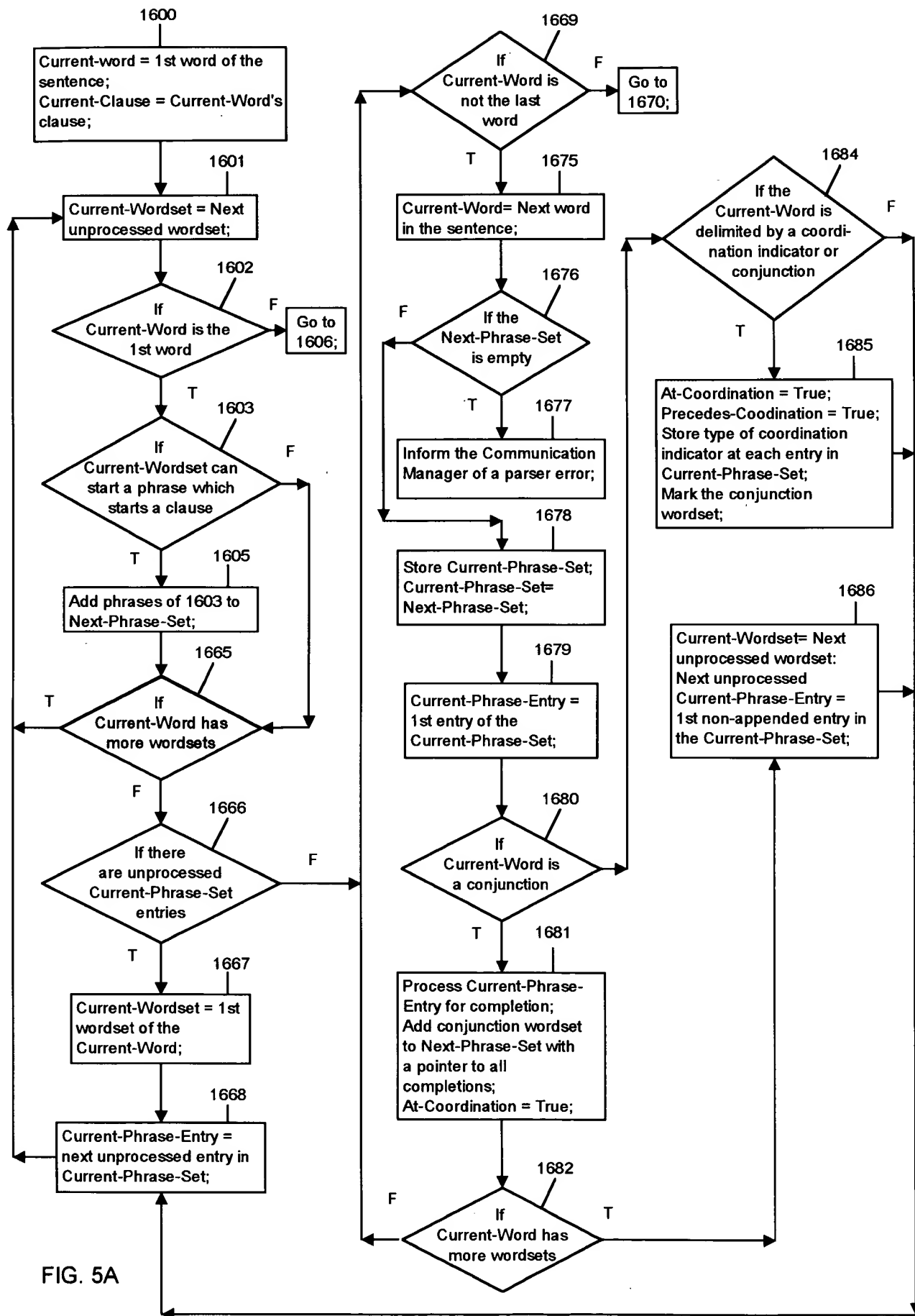


FIG. 4B



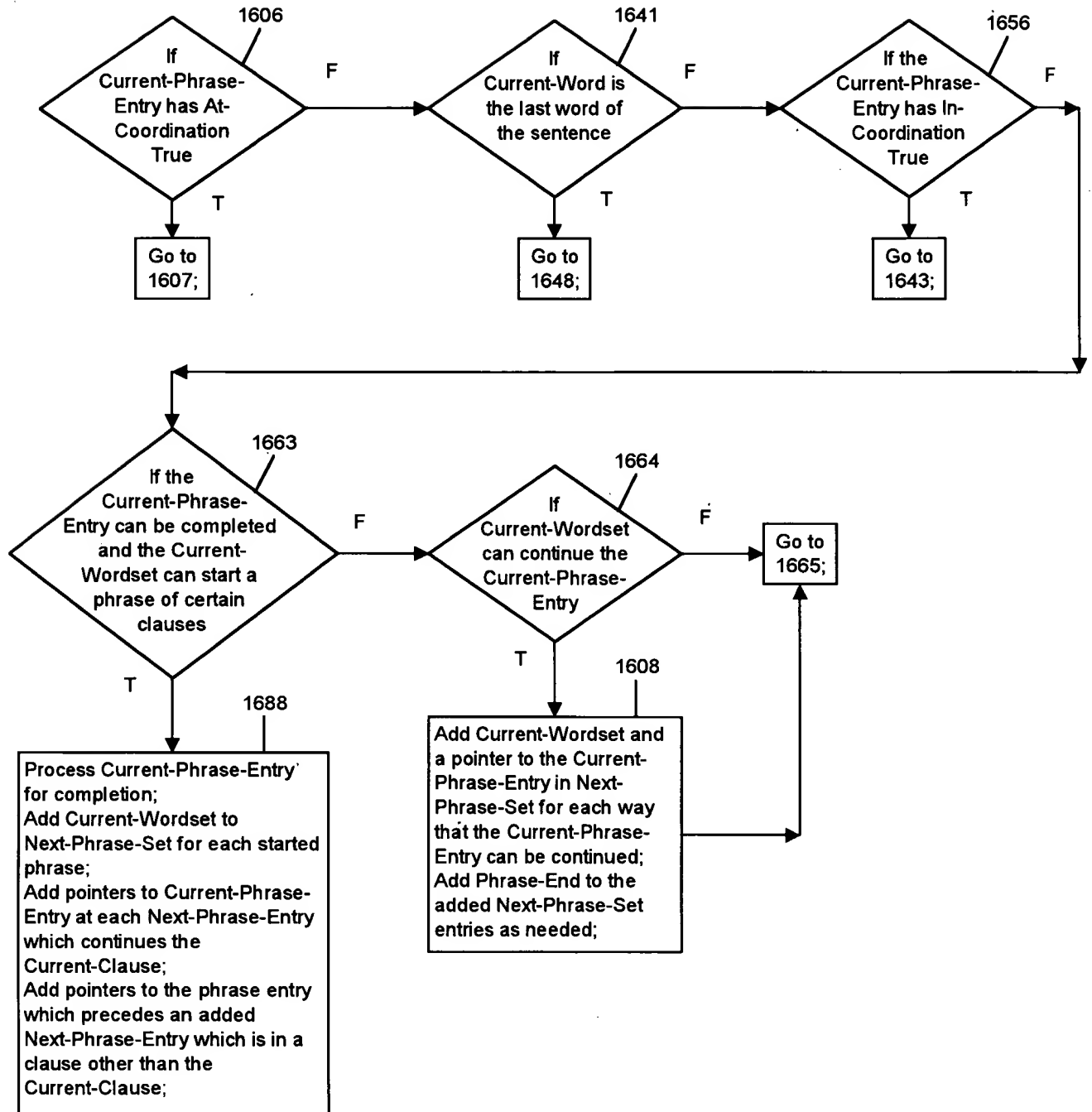


FIG. 5B

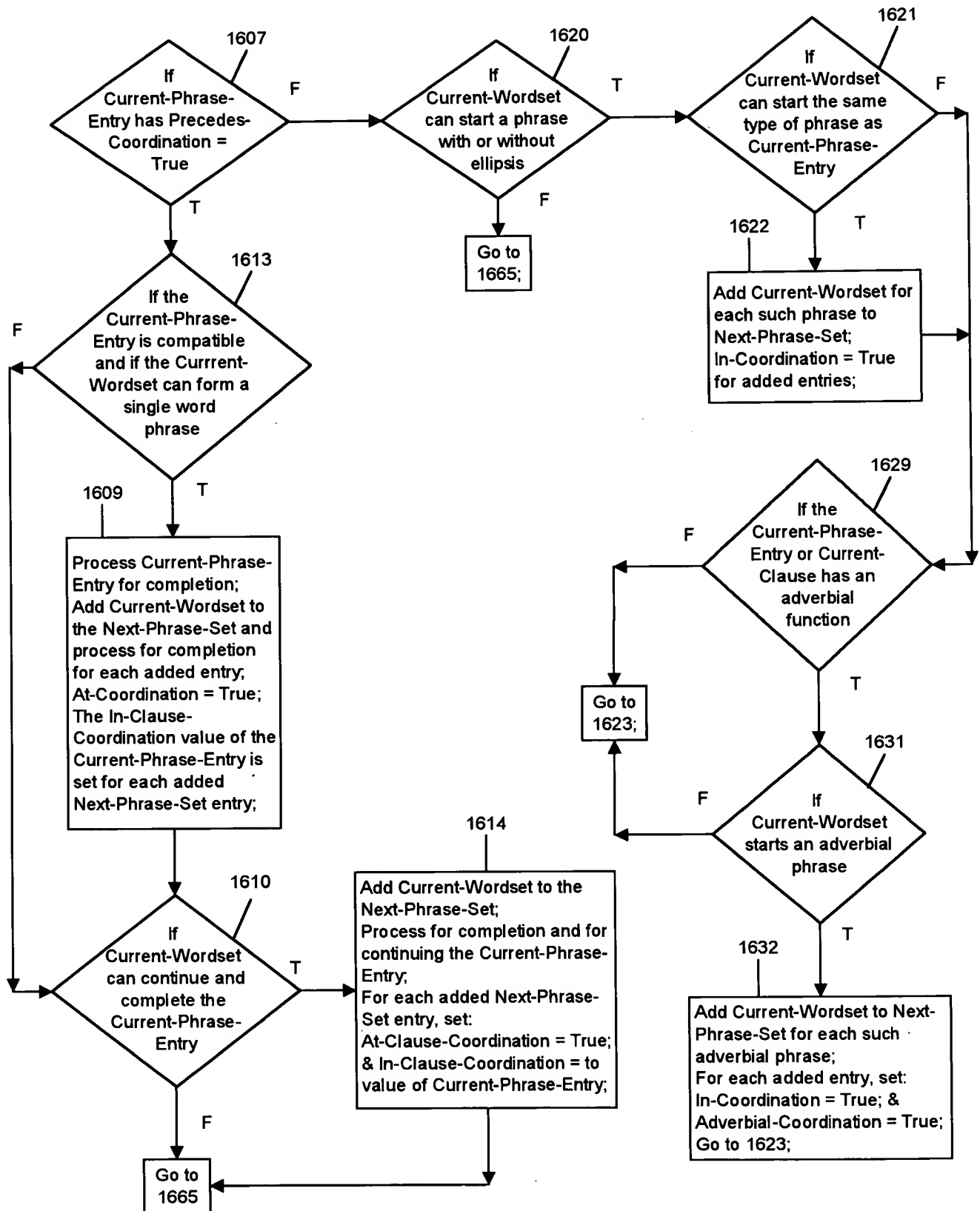


FIG. 5C


```

graph TD
    1623{1623  
If  
Current-Wordset  
can start a  
clause}
    1624{1624  
If  
Current-Wordset  
starts a main  
clause}
    1625{1625  
If  
Current-Clause  
is  
completable}
    1626[1626  
Mark all main clause  
starts as Coordinated-  
Clause-Start;]
    1627[1627  
Mark all main clause  
starts as Interpolated-  
Clause-Start;]
    1628{1628  
If  
Current-Wordset  
starts a subordinate  
clause}
    1637[1637  
Mark all subordinate  
clause starts which  
continue the main clause  
as  
Continuing-Subordinate-  
Clause-Start;  
Mark all other subordinate  
clause starts as  
Separate-Subordinate-  
Clause-Start;]
    1639[1639  
For each clause start: Add  
Current-Wordset to  
Next-Phrase-Set including  
the marked information;  
For each added entry: set  
In-Coordination = True; &  
In-Clause-Coordination  
= True;]
    1634[1634  
Go to  
1634;]

    1623 -- F --> 1634
    1623 -- T --> 1624
    1624 -- F --> 1628
    1624 -- T --> 1625
    1625 -- T --> 1626
    1625 -- F --> 1634
    1626 --> 1627
    1627 --> 1628
    1628 -- T --> 1637
    1628 -- F --> 1639
    1637 --> 1639
    1639 --> 1634
  
```

FIG. 5D

```

graph TD
    1634{1634  
If the  
Current-Wordset  
is incomplete and if the  
Current-Preceding-  
Clause is incomplete  
in the same way}
    1635[1635  
Addition-Preceding-  
Clauses = False;]
    1636[1636  
Add the Current-Wordset to the  
Next-Phrase-Set for each phrase  
which continues the  
Current-Clause  
and the Current-Preceding-Clause;  
Include a pointer to the Current-  
Phrase-Entry and the last phrase  
entries of the Current-Preceding-  
Clause which are completable and  
which allow the Current-Wordset to  
start a continuing phrase;  
For each added entry, set:  
Interpolated-Clause-Continuation =  
True;  
In-Coordination = True; &  
In-Clause-Coordination = True;]
    1638{1638  
If there  
are remaining  
entries added at 1636  
and if there is another clause  
preceding the Current-  
Preceding-Clause, and if that  
preceding clause is incomplete  
in the same way as the  
Current-Clause}
    1640[1640  
Current-Preceding-Clause  
= Next preceding clause;  
Additional-Preceding-  
Clauses = True;]
    1665[1665  
Go to  
1665;]

    1634 -- F --> 1665
    1634 -- T --> 1635
    1635 --> 1636
    1636 --> 1638
    1638 -- F --> 1665
    1638 -- T --> 1640
    1640 --> 1636

```

FIG.5E

```

graph TD
    1643{1643  
If the  
Current-Wordset  
continues the  
Current-Phrase-  
Entry}
    1645[1645  
Add the Current-Wordset to  
the Next-Phrase-Set for each  
way that the Current-Wordset  
can continue the  
Current-Phrase-Entry;  
Include a pointer to the  
Current-Phrase-Entry;  
For each added entry:  
In-Coordination = True;]
    1644{1644  
If the  
Current-Phrase-  
Entry is a completed  
adverbial phrase without a  
succeeding modifiee and the  
Current-Wordset can  
start a noun  
phrase}
    1647{1647  
If  
Current-Phrase-  
Entry is a completable  
noun phrase and the  
Current-Wordset starts  
an adverbial  
phrase}
    1646[1646  
Add an entry to the Current-  
Phrase-Set for each way the  
Current-Phrase-Entry can be  
completed;  
Add the Current-Wordset to  
the Next-Phrase-Set for each  
started phrase including a  
pointer to entries added to the  
Current-Phrase-Set;  
For each added entry: set  
In-Coordination = True; &  
Adverbial-Coordination = True;]
    1653[1653  
Process each completable  
phrase for completion;  
For each continuing phrase  
start: Add the Current-Wordset  
to the Next-Phrase-Set with a  
pointer to each completion  
which has its clause continued;  
For each added entry: set  
In-Coordination = True;]
    1651{1651  
If the  
Current-Phrase-  
Entry is completable  
and the Current-Wordset  
can start a phrase which  
continues a clause in  
the sentence}
    1654{1654  
If the  
Current-  
Phrase-Entry has  
In-Coordination  
True}
    1655[1655  
For all entries added at  
1645, 1646, and 1653,  
set:  
In-Clause-Coordination =  
True;]
    G1665_1[Go to  
1665;]
    G1665_2[Go to  
1665;]

    1643 -- F --> 1644
    1643 -- T --> 1645
    1645 --> 1644
    1644 -- F --> 1647
    1644 -- T --> 1646
    1647 -- T --> 1653
    1647 -- F --> 1651
    1651 -- F --> G1665_1
    1651 -- T --> 1653
    1653 --> 1654
    1654 -- T --> 1655
    1654 -- F --> G1665_2
    1655 --> G1665_2
  
```

FIG. 5F

```
graph TD
    1648{1648  
If the  
Current-Phrase-Entry is completable  
and continues its Current-Clause and the Current-Wordset forms a complete phrase which ends the Current-Clause or another clause}
    1649{1649  
If the  
Current-Wordset can complete the Current-Phrase-Entry and the completed Current-Phrase-Entry can complete the Current-Clause}
    1690[1690  
Process each completable phrase for completion;  
Add the Current-Wordset to the Next-Phrase-Set with a pointer to all completable phrases which have their clauses continued by the added entry;  
For each added entry, set: Clause-End = True; & Mark all types of clause completions True;]
    1650[1650  
Add the Current-Wordset to the Next-Phrase-Set for each possible completion and process each added entry for completion;  
For each added entry, set: Clause-End = True; & Mark all types of clause completions True;  
Go to 1665;]
    1665[Go to 1665;]

    1648 -- T --> 1690
    1648 -- F --> 1649
    1649 -- T --> 1650
    1649 -- F --> 1665
    1690 --> 1649
    1650 --> 1649
```

FIG. 5G

```
graph TD; 1670{1670  
If the Next-Phrase-Set  
is empty} -- T --> 1672[1672  
Inform the Communication  
Manager of a Parser  
failure;]; 1670 -- F --> 1673{1673  
If the Next-Phrase-Set  
has one entry}; 1673 -- T --> 1673[1673  
Invoke Dictionary Look Up  
Step 18;]; 1673 -- F --> 1674[1674  
Inform the Communication  
Manager of multiple syntax  
interpretations;];
```

FIG. 5H

PRONOUN	REFERENT PROPERTIES	CONFIDENCE LEVEL	SPECIAL GRAMMAR FUNCTIONS
"DO"	Verb; :	2	
"I"	Noun, 1st person, singular; Subjective, UNIQUE; :	4 4	FN25
"IT"	Noun, thing, singular; Noun, SPECIAL MEANING; Noun, clause, CATAPHORIC; Noun, sentence, CATAPHORIC :	4 1 1 1	FN29, FN31...
"SOME"	Noun, person; Noun, thing; Noun, place; Noun, time; :	1 1 1 1	FN892 FN892 FN892 FN892
"THAT"	Noun, thing, singular or noncount; Noun, person, singular Noun, SPECIAL MEANING; Noun, clause; Noun, sentence; :	1 1 1 1 1	FN77 FN225 FN37
"TODAY"	Noun, UNIQUE; :	4	FN110
NUMBERS	Noun, number; Noun, countable; Noun, SPECIAL MEANING; Adjective; :	4 2 4 4	FN 110, FN111 FN112

FIG. 6A

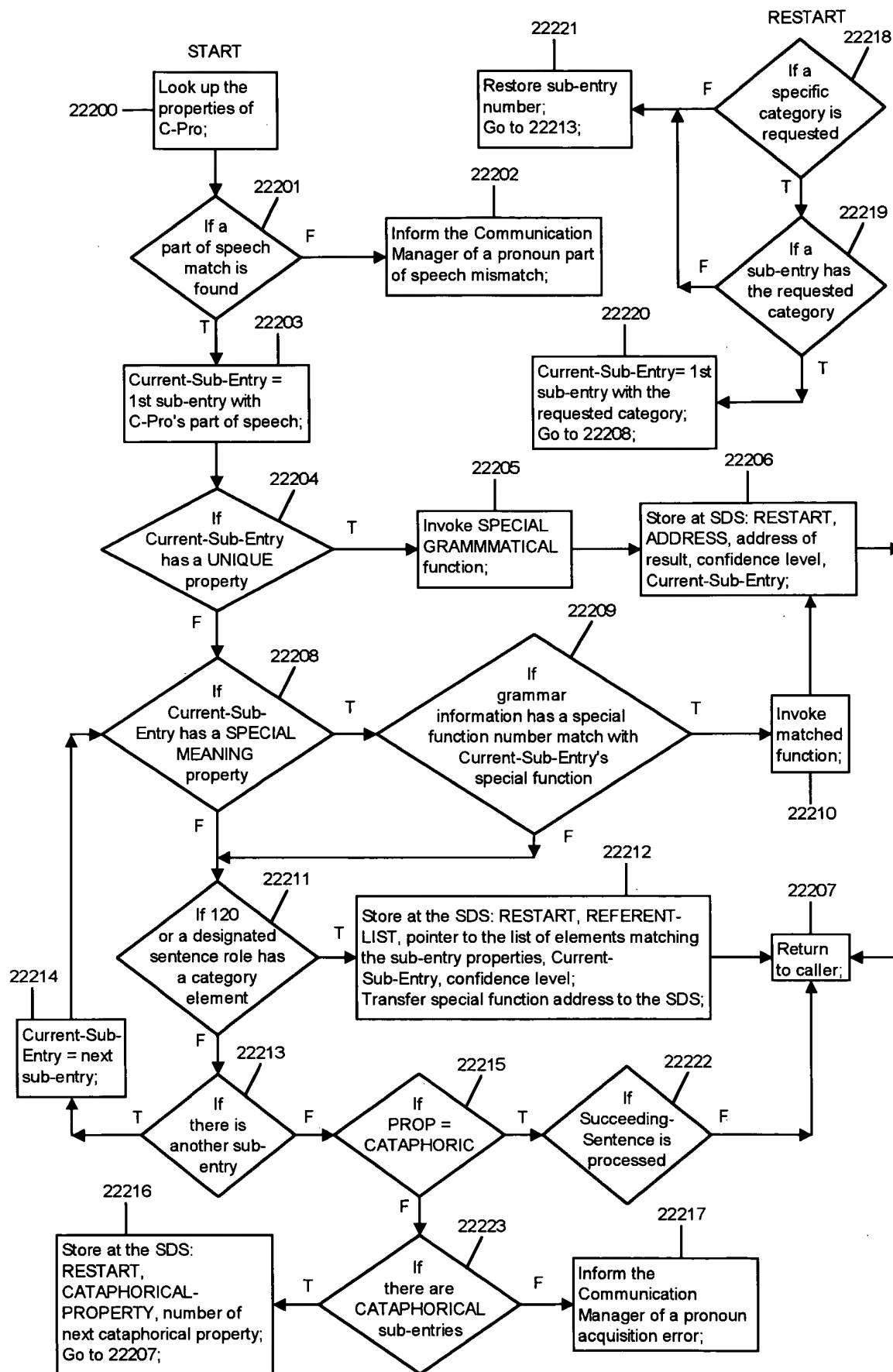


FIG. 6B

TEXT (single adjective)	MULTI-WORD FUNCTION LIST	DEFAULT REFERENCE TYPE	FUNCTION TYPE	FUNCTION ADDRESS and ASSOCIATED PARAMETERS	CONFIDENCE LEVEL	NEXT FUNCTION ADDRESS
Text of adjective	Set of multi-word symbols and associated starting addresses of its definition	Value of: SPECIFIC, SPECIFIC UNKNOWN, GENERAL	Value of: SELECTION, QUANTIZATION, etc.	A set of function addresses and parameters pairs which correspond to the adjective function	Number between 1 and 4 with 4 being the highest confidence level	NULL or the starting address of the next alternative function definition

FIG. 7A

MULTI-WORD SYMBOL	DEFAULT REFERENCE TYPE	LIST OF FUNCTION ADDRESSES and ASSOCIATED PARAMETERS OR POSITIONS	CONFIDENCE LEVEL	NEXT FUNCTION ADDRESS
Symbol used to look up functions associated with a function word position	Value of: SPECIFIC, SPECIFIC UNKNOWN, GENERAL	Ordered list of elements; Each element is either an address of a function, its type, and other parameters, or a position location for accessing the function word in the SDS	Number between 1 and 4 with 4 being the highest confidence level	NULL or the starting address of the next alternative function definition

FIG. 7B

```

graph TD
    22300[RESTART = 22301;  
Store RESTART in SDS;  
22300] --> 22301{If  
function address  
is given  
22301}
    22301 -- F --> 22302[Function-Definition-  
Address = 1st  
definition address;  
22302]
    22301 -- T --> 22303[Function-Definition-  
Address = given  
definition address;  
22303]
    22302 --> 22304{If  
definition has  
a default  
reference  
type  
22304}
    22303 --> 22304
    22304 -- F --> 22301
    22304 -- T --> 22305{If  
120 has the  
modified  
referent  
22305}
    22305 -- F --> 22306[Store at SDS: NEW-  
REFERENCE,  
default referent type;  
22306]
    22305 -- T --> 22307{If the  
default  
reference type  
is the same  
as in 120  
22307}
    22307 -- F --> 22308[Store at SDS: OLD-  
REFERENCE,  
pointer to the  
modified noun in  
120;  
22308]
    22307 -- T --> 22309{If the  
function word  
is modified by  
a degree  
adverb  
22309}
    22309 -- F --> 22301
    22309 -- T --> 22310[Degree-Mult =  
Product of  
multiplying degree  
adverb degree  
numbers;  
22310]
    22310 --> 22311{If  
there is a  
function to be  
processed  
22311}
    22311 -- F --> 22312[Store at the SDS:  
confidence level, next  
definition address  
or NULL;  
Return to caller;  
22312]
    22311 -- T --> 22313{If the  
definition is  
for a multi-word  
phrase  
22313}
    22313 -- F --> 22301
    22313 -- T --> 22314[Current-Function =  
function at definition  
address or function  
selected with the  
definition's symbol at  
the position's adjective;  
22314]
    22314 --> 22315{If  
Current-  
Function is for  
selection  
22315}
    22315 -- F --> 22316{If  
Current-  
Function is for  
quantization  
only  
22316}
    22315 -- T --> 22317[Evaluate a compatible,  
possible function;  
Store and identify the  
selection results or the  
function address in the  
SDS;  
22317]
    22316 -- F --> 22318{If  
Current-  
Function is for  
quantization  
only  
22318}
    22316 -- T --> 22317
    22318 -- F --> 22320[Go to  
22320;  
22320]
    22318 -- T --> 22319[Evaluate a compatible,  
possible function;  
Store and identify the  
quantization results or the  
function address in the  
SDS;  
22319]
    22317 --> 22340{If  
definition has  
a compatible  
function  
22340}
    22319 --> 22340
    22340 -- F --> 22342{If  
there is another  
definition  
22342}
    22340 -- T --> 22344[Function-Definition-  
Address = Next  
definition address;  
Go to 22304;  
22344]
    22342 -- F --> 22346[Go to  
AF-Fail;  
22346]
    22342 -- T --> 22344
  
```

FIG. 7C

```

graph TD
    22320{22320  
If  
Current-Function  
is a comparative  
function  
only}
    22322{22322  
If  
Current-Function  
is a quantization  
or comparative  
function}
    22324{22324  
If  
Current-Function  
is an inclusion  
function}
    22321[22321  
Evaluate a compatible,  
possible comparative function;  
Identify and store the  
comparison data in the SDS;}
    22323[22323  
Evaluate compatible, possible  
functions;  
Store in the SDS:  
AMBIGUOUS-QUANTIZATION/  
COMPARISON-FUNCTION,  
the quantization type and  
value, the comparison type  
and value;}
    22325[22325  
Store and identify a  
compatible inclusion  
function and related  
information in the SDS;}
    22326{22326  
If  
Current-Function  
is an exclusion  
function}
    22328{22328  
If  
Current-Function  
is a degree adverb  
function}
    22330{22330  
If  
Current-Function  
is a special  
function}
    22327[22327  
Store and identify a  
compatible exclusion  
function and related  
information in the SDS;}
    22329[22329  
Multiply a numerical  
quantity, and store the  
result in the SDS; or store  
Degree-Mult, and degree  
number or equivalent in  
the SDS;}
    22331[22331  
Evaluate a compatible,  
possible special function;  
Store and identify the  
result or the function  
address in the SDS;}
    22332{22322  
If  
Current-Function  
is a negative  
function}
    22333[22333  
Evaluate a compatible,  
possible negative function;  
Store and identify the  
result or the function  
address in the SDS;}
    22340[Go to  
22340]

    22320 -- T --> 22321
    22320 -- F --> 22322
    22322 -- T --> 22323
    22322 -- F --> 22324
    22324 -- T --> 22325
    22324 -- F --> 22326
    22321 --> 22340
    22323 --> 22340
    22325 --> 22340
    22326 -- T --> 22327
    22326 -- F --> 22328
    22327 --> 22340
    22328 -- T --> 22329
    22328 -- F --> 22330
    22329 --> 22340
    22330 -- T --> 22331
    22330 -- F --> 22332
    22331 --> 22340
    22332 -- T --> 22333
    22332 -- F --> 22340
    22333 --> 22340

```

PREPOSITION TEXT REPRESENTATION	RELATION TYPES	CONFIDENCE LEVEL
Preposition text, representation number	List of A-Relation designations, S-Relation designations, and/or T-Relation designations	Confidence level for each designation

A-Relation descriptor associated with a designation:

Type: Partitive, Possessive, Function, or Group;
Relation Characteristic;
Preceding/Succeeding Pointers;
Memory 100 Pointers;
Specific Information;
Designation;

S-Relation descriptor associated with a designation:

Source Requirement Descriptor;
Destination Requirement;
Destination: MODIFIEE or COMPLEMENT;
Relation Setting Function;
Designation;

T-Relation descriptor associated with a designation:

Destination: MODIFIEE or COMPLEMENT;
Relation Setting Function;

FIG. 8A

```

graph TD
    2200[RESTART = 2201;  
Store at SDS:  
RESTART,  
value of RESTART;  
Next-Relation-Address  
= 1st address of  
Cur-Prep;  
2200] --> 2201[Current-Relation =  
Relation at Next-  
Relation-Address;  
Next-Relation-Address  
= Address of next  
relation or NULL if  
there is none;  
CN-Prep-Status =  
NULL;  
2201]
    2201 --> 2202{If  
Current-  
Relation is an  
A-Relation  
2202}
    2202 -- F --> 2201
    2202 -- T --> 2204[Search for the  
A-Relation in 120 or 90;  
2204]
    2204 --> 2209{If  
Current-  
Relation is in  
120 or 90  
2209}
    2209 -- T --> 2203[Store found relation in  
120 if its not there;  
2203]
    2209 -- F --> 2205{If  
Current-  
Relation is a  
group relation  
2205}
    2205 -- T --> 2208[Generate the group  
relation;  
2208]
    2205 -- F --> 2206{If  
there is another  
relation  
2206}
    2208 --> 2203
    2203 --> 2207[Store at the SDS: relation  
type, relation address in 120  
or 90, the confidence level,  
Next-Relation-Address;  
CN-Prep-Status = FOUND;  
2207]
    2207 --> 2217[Return to  
caller;  
2217]
    2206 -- F --> 2218[Store found T-Relation  
in 120 if needed, or  
generate and store the  
T-Relation in 120;  
2218]
    2206 -- T --> 2201
    2218 --> 2207
    2210{If  
Current-  
Relation is an  
S-Relation  
2210} -- F --> 2201
    2210 -- T --> 2211[Search for the S-Relation  
in 120 or 90, or determine  
if one can be generated;  
2211]
    2211 --> 2219{If the  
S-Relation  
is in 120 or 90,  
or can be  
generated  
2219}
    2219 -- T --> 2212[Store a relation found in  
120 at 90; or  
Generate a new relation  
and store in 120 if none is  
found;  
Go to 2207;  
2212]
    2219 -- F --> 2213{If an  
incompatible  
S-Relation  
was found  
2213}
    2213 -- T --> 2214[Store in SDS:  
INCOMPATIBLE-  
S-Relation, address  
of the incompatible  
relation;  
2214]
    2213 -- F --> 2215[Search for the  
T-Relation in 120 or  
90, or determine if one  
can be generated;  
2215]
    2214 --> 2216{If a  
T-Relation was  
found or can be  
generated  
2216}
    2216 -- T --> 2218
    2216 -- F --> 2215
    2215 --> 2216
  
```

FIG. 8B



PREPOSITION TEXT REPRESENTATION	RELATION TYPES	CONFIDENCE LEVEL
Preposition text, representation number	List of AMF-Relation designations, F-Relation designations, T-Relation designations, and/or P-Relation designations	Confidence level for each designation

AMF and F -Relation descriptor associated with a designation:

Type: AMF: Partitive, Possessive, Function, or Group; F: Function;
Relation Characteristic;
Preceding/Succeeding Pointers;
Memory 100 Pointers;
Specific Information;
Designation;

T-Relation descriptor associated with a designation:

**Destination: SUBJECT or COMPLEMENT;
Relation Setting Function;
Default state(s) and/or properties;
Designation;**

P-Relation descriptor associated with a designation:

**Purpose Relation Type;
Designation;**

FIG. 8C

```

graph TD
    2220[2220  
RESTART = 2221;  
Store at SDS:  
RESTART,  
value of RESTART;  
Next-Relation-Address  
= 1st address of  
Cur-Prep;] --> 2228{2228  
If  
Cur-Prep  
can be an  
adverbial}
    2228 -- T --> 2230[2230  
Invo-Mod-Set = Subject;  
Invo-ADJ = Adjective;  
60-Start = 60872;  
ADJ-PREP-Return = 2236;  
ADJ-PREP-Status = False;  
Call 60[60-Start, Invo-Mod-Set,  
Invo-ADJ, ADJ-PREP-Status,  
ADJ-PREP-Return];]
    2228 -- F --> 2236{2236  
If  
ADJ-PREP-Status =  
Found-In  
90}
    2236 -- T --> 2245[2245  
Set adjective to modify the  
subject;  
Current-Relation = NULL;  
ADJ-PREP-Status =  
Completed;]
    2236 -- F --> 2221[2221  
Current-Relation = Relation  
at the  
Next-Relation-Address;  
Next-Relation-Address =  
Next relation address or  
NULL if there is none;]
    2221 --> 2227{2227  
If  
Current-Relation is a  
T-Relation}
    2227 -- F --> 2229[Go to  
2229;]
    2227 -- T --> 2231{2231  
If the  
subject and  
complement  
are both  
concrete  
nouns}
    2231 -- F --> 2233{2233  
If the  
subject and  
complement  
are clauses or  
clause  
equivalents}
    2231 -- T --> 2232[2232  
Search for a T-Relation  
in 120 or 90;  
If none is found,  
determine if one can be  
generated;]
    2232 --> 2234{2234  
If a  
T-Relation  
was found or is  
generatable}
    2233 -- T --> 2235[2235  
Search for a T-Relation  
in 120 or 90;  
If none is found, call 70  
to determine if one can  
be generated;]
    2233 -- F --> 2225{2225  
If  
Next-Relation-Address is  
not NULL}
    2234 -- T --> 2226[2226  
Store in the SDS:  
relation type, relation  
address, computed  
confidence level,  
Next-Relation-Address;  
ADJ-PREP-Status =  
COMPLETED;]
    2234 -- F --> 2225
    2235 --> 2226
    2225 -- T --> 2226
    2225 -- F --> 2222[2222  
Return to  
caller;]
    2226 --> 2222
    2222 --> 2226

```

FIG. 8D

```

graph TD
    2229{2229  
If Current-Relation is an AMF-Relation}
    2249{2249  
If Current-Relation is a P-Relation}
    2250{2250  
If the complement is a concrete noun}
    2237{2237  
If the subject and complement are both concrete nouns and the adjective can modify concrete nouns}
    2246{2246  
If the subject is a clause or equivalent}
    2257{2257  
If the clause complement is a purpose relation clause with a Current-Relation purpose type}
    2238[2238  
Search for an AMF-Relation in 120 or 90;]
    2239{2239  
If an AMF-Relation is found}
    2240[2240  
Search for a noun contained in the AMF-Relation which can be modified by the adjective;]
    2241{2241  
If a noun is found}
    2242[2242  
Store in the SDS: the adjective, the found noun, and a processing symbol;  
Store the AMF-Relation and the found noun as needed in 120;  
Store Modal-V at the complement's SDS position;]
    2251[2251  
Search for a function relation containing the complement possibly with an AMF-Relation such that the function relation clause is a purpose relation clause of the adjective with the Current-Relation purpose type in 120 or 110;]
    2252{2252  
If a clause is found}
    2258[2258  
Associate the found clause with the adjective in 120 as needed;  
Store in the SDS: a pointer to the clause, a processing symbol, and Modal-V;]
    2225[2225  
Go to 2225;]
    2243[2243  
Go to 2243;]

    2229 -- T --> 2237
    2229 -- F --> 2249
    2237 -- T --> 2238
    2237 -- F --> 2225
    2238 --> 2239
    2239 -- T --> 2240
    2239 -- F --> 2225
    2240 --> 2241
    2241 -- T --> 2242
    2241 -- F --> 2225
    2242 --> 2225
    2249 -- T --> 2250
    2249 -- F --> 2246
    2250 -- T --> 2251
    2250 -- F --> 2257
    2251 --> 2252
    2252 -- T --> 2258
    2252 -- F --> 2225
    2257 -- T --> 2258
    2257 -- F --> 2225
    2246 -- T --> 2225
    2246 -- F --> 2243
    2225 --> 2222[2222]
    2243 --> 2222
    2258 --> 2222
    2242 --> 2222
  
```

FIG. 8E

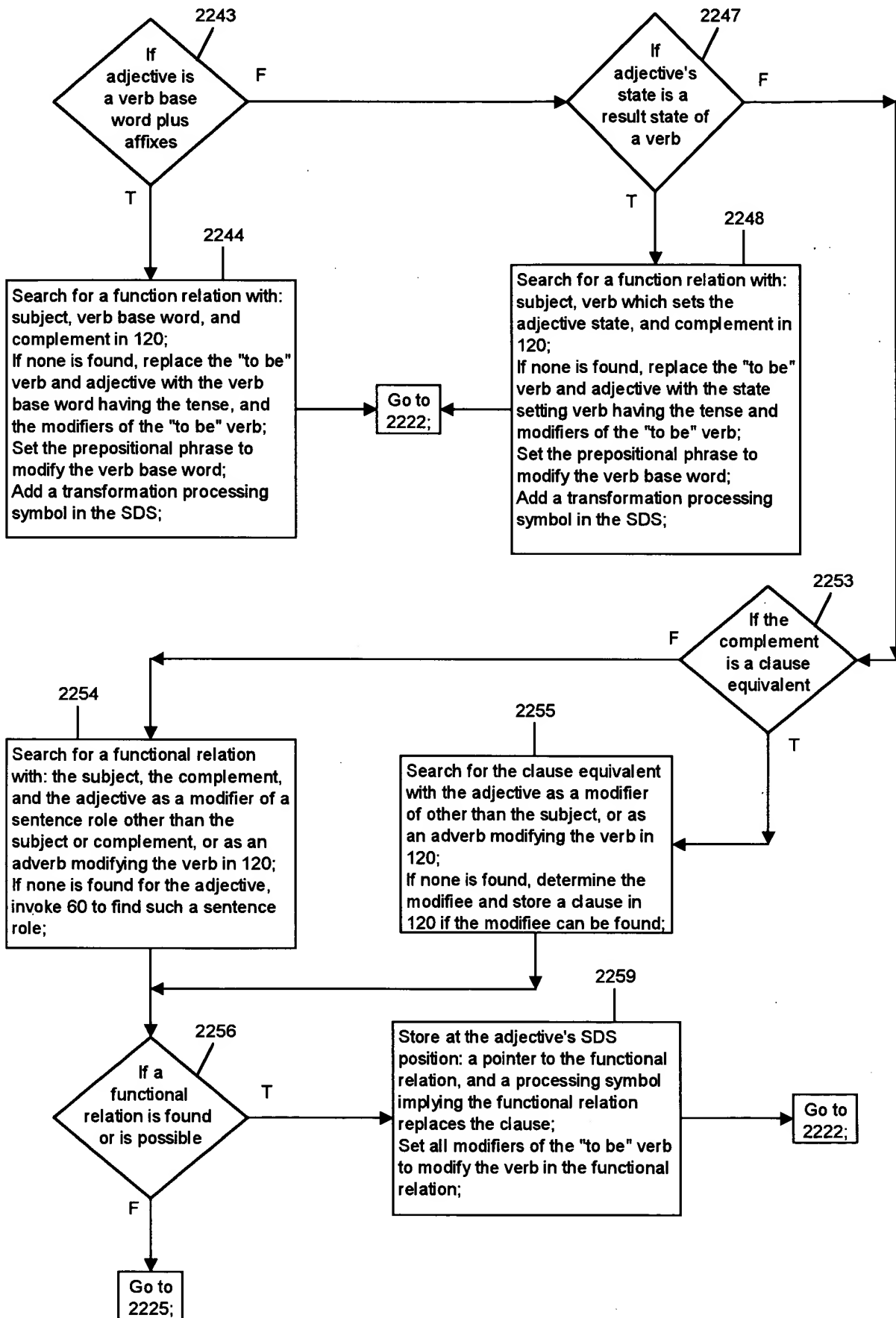


FIG. 8F

002250-03512960


MODIFYING ADVERBIAL SUBCLASS DATA STRUCTURE
SOURCE DESCRIPTOR: Semantic role; Required value or value range, and units; Required state(s) and/or property(s); Function to obtain required state(s) and/or property(s); Required parameters;
DESTINATION DESCRIPTOR: Requirements of Modifiee; or NULL set;
Function of Adverbial Subclasses Include: Set adverbial subclass values; Select modifiee word sense or word sense component; Initiate processes;

FIG. 9A

**Current-Adverbial = Invocation
adverbial;
Delete SDS information of a previously
processed Current-Adverbial;
RESTART = 2287;
Store in the SDS: RESTART, and its
value;
WILD = False;**

Find all matches of the modifier's adverbial semantic roles with Current-Adverbial's semantic roles; Store all matches in the Current-Evaluation-Set; Currently-Most-Recently-Used-Semantic-Role = most recently used semantic role in 120;

- 2273

T 

If a
semantic
role match
is found

If
there is
an untried
semantic role

2282

Mark all entries without FAILED in the Current-Evaluation-Set as UNTRIED;
 Current-Most-Recently-Used-Semantic-Role = next untried semantic role in 120;

Determine if the source and destination descriptors of the Current-Match are satisfied;

If WILD is False

T

Mark all entries without FAILED in the Current-Evaluation-Set as UNTRIED;
Current-Most-Recently-Used-Semantic-Role = MATCHES-ANY-SEMANTIC-ROLE-VALUE;
WILD = True;

If descriptors are satisfied

evaluate evaluable Current-Match
 actions;
 are in the SDS: results, addresses,
 position of modifiee's matched adverbial
 class, pointer to the Current-
 evaluation-Set, pointer to the Current-
 match, pointer to the Currently-Most-
 recently-Used-Semantic-Role, WILD;

2277

**Mark Current
as FAILED;**

RESTORE
Restore st
from SDS

2278

T

UNTRIED
entries in Current-
Evaluation-
Set

Go to
2279;

2282

**Return to
Caller;**

2285

Store a TRIED-BUT-FAILED symbol in the SDS:

FIG.9B

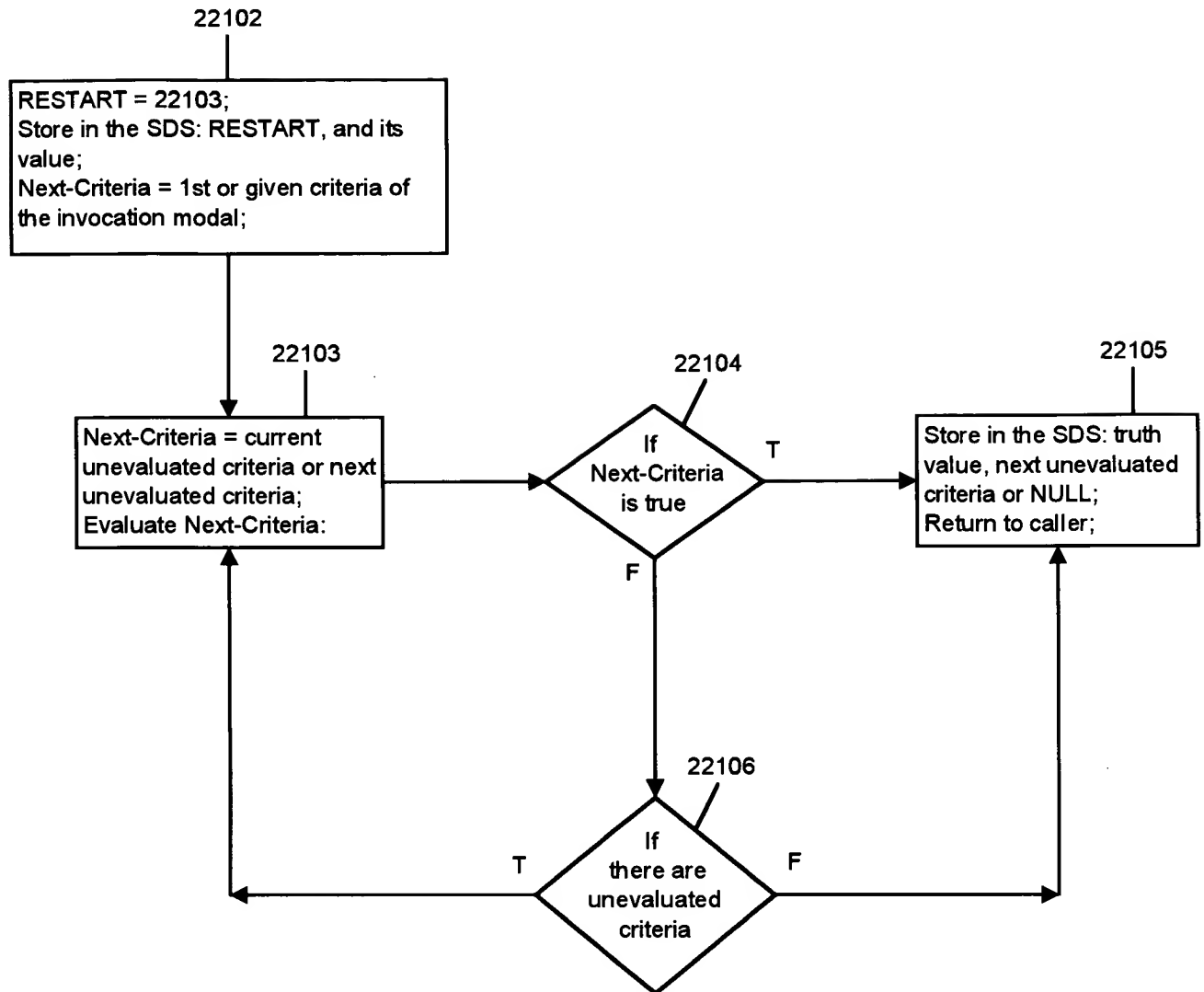


FIG. 10A

TENSE	VERB TYPE	TIME PERIODS WHEN TRUE			TIME SHIFTS TO:	USAGE SITUATION
PRESENT	STATIVE EVENTIVE HABITUAL	Before > not Before Before >	Now Now Now	< After not After < After	PAST FUTURE PAST FUTURE PAST FUTURE	us1 us2,us3,us4,us5,us6 us8
PAST	STATIVE EVENTIVE HABITUAL	Before Past not Before Past Before Past	Past Time Point Past Time Point Past Time Point	After Past not After Past After Past	PRESENT to - PAST and - FUTURE to PAST	us9 us9 us9
FUTURE	STATIVE EVENTIVE HABITUAL	not Now not Now not Now	Future Future Future	beyond Future not beyond Fut. beyond Future	none none none	us10,us11 us10,us11 us10,us11
PERFECTIVE TENSES						
PRESENT	STATIVE EVENTIVE HABITUAL	Past recent Past Past	Present Present Present	possibly Future possibly Future possibly Future	none none none	us12 us12 us12
PAST	STATIVE EVENTIVE HABITUAL	Past Period Past Period Past Period	not Now not Now not Now	not Future not Future not Future	none none none	us13 us13 us13
FUTURE	STATIVE EVENTIVE HABITUAL	possibly Past possibly Past possibly Past	Now Now Now	Expected to - be true in - the Future	none none none	us14 us14 us14
PROGRESSIVE TENSES						
PRESENT, PAST, and FUTURE	STATIVE EVENTIVE HABITUAL	not Before just Before not Before	near Time Point Time Point near Time Point	not After just After not After	none none none	us15,spec. meaning us16 us15
PERFECTIVE PROGRESSIVE TENSES						
PRESENT	EVENTIVE HABITUAL	Before Now Before Now	Now near Now	possibly Future possibly Future	none none	us 17 us 17
PAST	EVENTIVE HABITUAL	Past Past	not Now not Now	not Future not Future	none none	us18 us18
FUTURE	EVENTIVE HABITUAL	possibly Past possibly Past	Now Now	Expected to - continue into - the Future	none none	us19 us19

1. *Chlorophyll a* (Chl *a*)

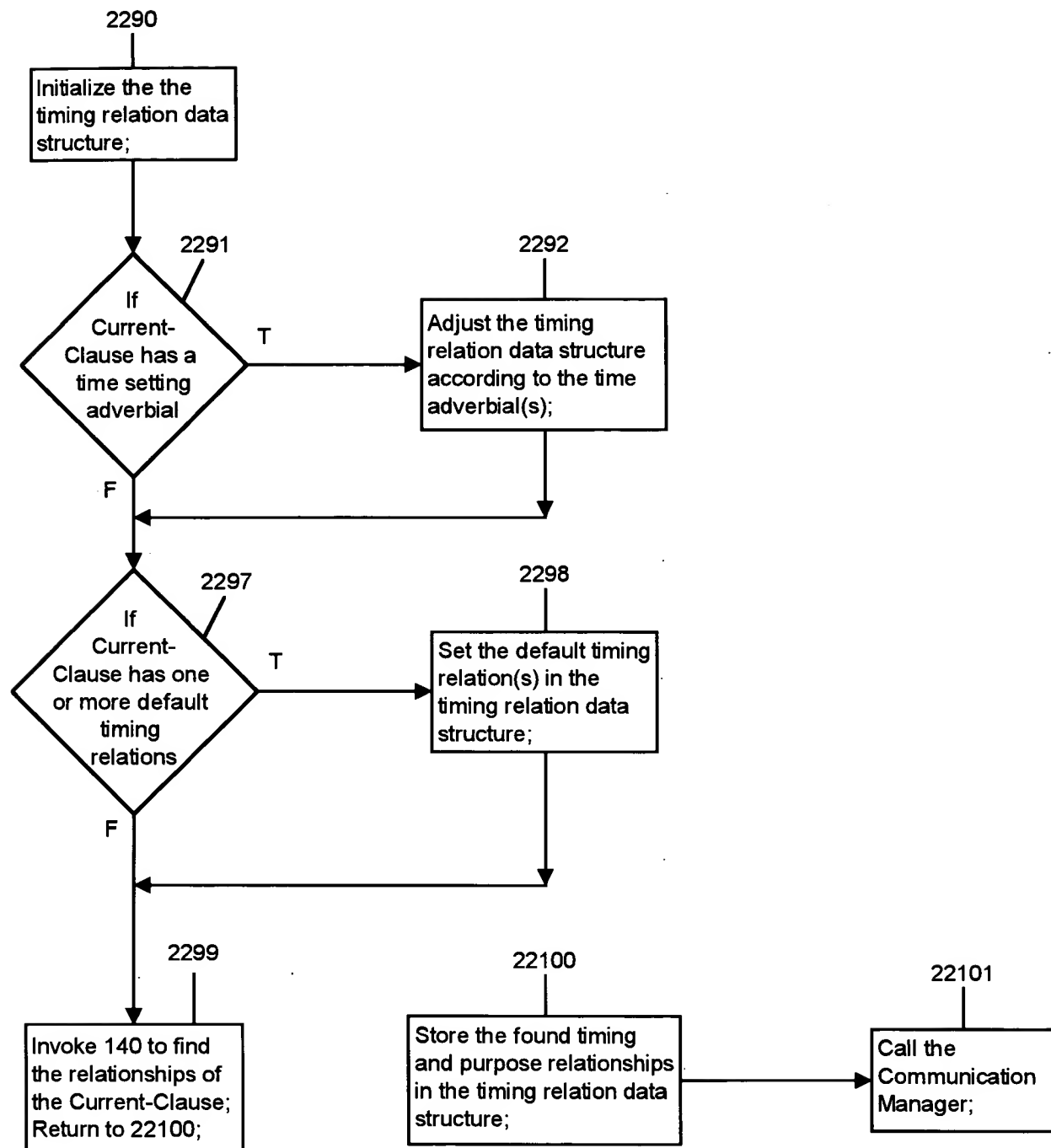


FIG. 10C

067180-0900

Conjunctions Joining Clauses

FIG. 11A

FIG. 11A

```

graph TD
    22110[22110  
Current-Conjunction = 1st conjunction in  
Cur-Conj-Set;  
Current-Conjunction-Elements = elements before  
Current-Conjunction;  
Current-Conjunction-Status = UNAMBIGUOUS;  
Current-Group = NULL;] --> 22111{22111  
If  
Current-  
Conjunction is  
not followed by an  
undelimited  
conjunction}
    22111 -- T --> 22112[22112  
Add element succeeding  
Current-Conjunction to  
Current-Conjunction-  
Elements;]
    22111 -- F --> 22114[22114  
Current-Element = 1st  
element succeeding  
Current-Conjunction;]
    22114 --> 22115{22115  
If  
Current-  
Element is in a  
group with  
preceding  
elements}
    22115 -- T --> 22116[22116  
Add Current-Element to  
Current-Conjunction-Elements;  
Mark Current-Element  
BY-INCLUSION;  
Current-Group = Containing group;]
    22115 -- F --> 22118[22118  
Add Current-Element to  
Next-Conjunction-Elements;  
Mark Current-Element BY-DEFAULT;  
Add a CONJUNCTION mark to the  
Current-Conjunction-Elements;]
    22116 --> 22115
    22118 --> 22120[22120  
Current-Conjunction-  
Status =  
AMBIGUOUS;]
    22120 --> 22122{22122  
If  
Current-  
Conjunction has  
single/multiple functions  
and there is an  
indicating word or  
phrase}
    22122 -- T --> 22123[22123  
Current-Function-Group =  
pointer to the function entry  
indicated for Current-  
Conjunction and the  
element type;]
    22122 -- F --> 22124[22124  
Current-Function-Group =  
pointer to all function  
entries for Current-  
Conjunction and the  
element type;]
    22123 --> 22125[22125  
Store in SDS: Current-Conjunction-  
Elements, Current-Function-Group,  
Current-Conjunction-Status,  
Current-Group;]
    22124 --> 22125
    22125 --> 22126{22126  
If there  
is another  
unprocessed  
conjunction}
    22126 -- T --> 22128[22128  
Current-Conjunction =  
Next unprocessed  
conjunction;]
    22126 -- F --> 22127[22127  
Return to  
caller;]
    22128 --> 22129{22129  
If  
Current-  
Conjunction-  
Status is  
AMBIGUOUS}
    22129 -- T --> 22131[22131  
Current-Conjunction-  
Elements = All  
elements preceding  
Current-Conjunction;]
    22129 -- F --> 22130[22130  
Current-Conjunction-  
Elements = Next-  
Conjunction-Elements  
and all other elements  
preceding Current-  
Conjunction;]
    22131 --> 22111
    22130 --> 22132[22132  
Current-Group =  
NULL;  
Current-Conjunction-  
Status =  
UNAMBIGUOUS;]
    22132 --> 22111

```

FIG. 11B

AFFIX TEXT	SOURCE	DESTINATION	TYPE NUMBER	FUNCTIONS	POINTER TO MODIFICATION ADVERBIAL SUBCLASSES FOR ADVERBIAL DESTINATIONS
"ly"	ADJECTIVE	ADVERB	1 2 3 . . .	F1 F2, F3 F7 . . .	P1 P2 P3 . . .
"ingly"	VERB	ADVERB	10 . . .	F18, F21, F99 . . .	P88 . . .

FIG. 12A

```

graph TD
    2400{2400  
If invocation  
descriptor  
contains  
GENERATE} -- T --> 2420[2420  
Look up the function-  
types which convert  
the SOURCE into the  
DESTINATION;]
    2400 -- F --> 2402[2402  
RESTART = 2408;  
Store at SDS:  
RESTART and its  
value, Function-Type-  
Set or NULL;]
    2420 --> 2421{2421  
If  
a function-  
type(s) is  
found}
    2421 -- T --> 2422[2422  
Function-Type-Set =  
Found function types;  
Store Function-Type-  
Set in the SDS;]
    2421 -- F --> 2423[2423  
Append in the SDS:  
FAIL;  
Return to caller;]
    2422 --> 2403{2403  
If  
invocation  
descriptor  
contains  
GENERATE}
    2403 -- T --> 2400
    2403 -- F --> 2404{2404  
If  
SDS contains  
a non-NULL  
Function-Type-  
Set}
    2404 -- T --> 2408[2408  
Current-Function = First  
unevaluated function-type  
in Function-Type-Set;]
    2404 -- F --> 2405[2405  
Look up the function-  
types corresponding to  
the SOURCE,  
DESTINATION and  
affixes;]
    2405 --> 2406{2406  
If  
a function-  
type(s) is  
found}
    2406 -- T --> 2407[2407  
Function-Type-Set =  
Found function-types;  
Store Function-Type-Set in  
the SDS;]
    2406 -- F --> 2415[2415  
Append in the SDS:  
RETURN-TO-INITIATING-  
CALLER;]
    2407 --> 2409{2409  
If  
invocation  
descriptor does not  
contain GENERATE  
and if Current-Function  
does not contain  
DELAY}
    2409 -- T --> 2410[2410  
Evaluate function(s) of  
Current-Function;]
    2409 -- F --> 2412[2412  
Append in the SDS:  
DELAYED-FUNCTION,  
address of Current-  
Function, evaluation  
address;]
    2410 --> 2411[2411  
Append in the SDS:  
RESULT-TYPE, result;]
    2411 --> 2414{2414  
If  
invocation  
descriptor has  
INVOCATION-  
RETURN}
    2412 --> 2414
    2414 -- T --> 2415
    2414 -- F --> 2416[2416  
Return to Step 18;]
    2415 --> 2415
  
```

FIG. 12B

```

graph TD
    2600{2600  
If  
Current-Sentence  
contains response  
ellipsis} -- T --> 2639[2639  
Go to 2640;  
(Response form  
ellipsis)]
    2600 -- F --> 2601{2601  
If  
there is  
unprocessed  
ellipsis}
    2601 -- F --> 2602[2602  
Res-Status =  
SUCCEED;  
Return to caller;]
    2601 -- T --> 2603[2603  
Current-Phrase = Next  
unprocessed phrase with  
ellipsis or next unprocessed  
ellipted phrase;]
    2603 --> 2604[2604  
Look up the type  
of ellipsis in the  
SDS;]
    2604 --> 2605{2605  
If  
there is  
general,  
coordination or  
comparative  
ellipsis}
    2605 -- T --> 2606{2606  
If the  
ellipted  
elements are  
known}
    2605 -- F --> 2699[2699  
ESUB = EOBJ = False;  
Go to 26100;  
(Nonfinite verb clause,  
verbless clause, and  
morphological word@  
ellipsis)]
    2606 -- T --> 2607{2607  
If  
there are  
untried known  
replacements}
    2606 -- F --> 2608[2608  
Current-Match =  
EXACT-MATCH;  
TPSV = All UNTRIED;  
First-Elliptical-Match =  
Current-Phrase;]
    2607 -- T --> 2613[2613  
RETURN = 2607;  
Current-Match =  
KNOWN;  
If not in the SDS, form  
TPSV for possible  
replacements;  
Set all vector elements  
to UNTRIED;  
Transfer 1st UNTRIED  
replacement to the  
SDS and set it to  
TRIED;]
    2607 -- F --> 2617[2617  
Go to 2617;]
    2608 --> 2610[2610  
Source-Phrase = Next  
UNTRIED phrase coordinated with  
Current-Phrase in the nearest  
preceding first order policy;  
Mark Source-Phrase to TRIED in  
the TPSV;  
RETURN = 2609;]
    2610 --> 2609{2609  
If  
Current-Phrase is  
coordinated}
    2609 -- T --> 2610
    2609 -- F --> 2622[2622  
Go to 2622;]
    2610 --> 2611{2611  
If an  
UNTRIED  
Source-Phrase  
is found}
    2611 -- T --> 2612{2612  
If  
Source-Phrase has a  
same phrase element  
Current-Match with  
Current-Phrase}
    2611 -- F --> 2622
    2612 -- T --> 2617
    2612 -- F --> 2610

```

FIG. 13A

```

graph TD
    2622[Source-Phrase = Next UNTRIED phrase with the same sentence role as Current-Phrase in the nearest preceding first order policy; Mark Source-Phrase to TRIED in the TPSV; RETURN = 2622;] --> 2623{If an UNTRIED Source-Phrase is found}
    2623 -- T --> 2624{If Source-Phrase has a same phrase element Current-Match or a same phrase Current-Match with Current-Phrase}
    2623 -- F --> 2625[Source-Phrase = Next UNTRIED phrase with the same phrase type as Current-Phrase in the nearest preceding first order policy; Mark Source-Phrase to TRIED in the TPSV; RETURN = 2625;]
    2624 -- T --> 2614{If next consecutive phrase after Current-Phrase is ellipted}
    2624 -- F --> 2627{If Source-Phrase has a same phrase element Current-Match or a same phrase Current-Match with Current-Phrase}
    2625 --> 2626{If an UNTRIED Source-Phrase is found}
    2626 -- T --> 2627
    2626 -- F --> 2628[Go to 2628;]
    2627 -- T --> 2614
    2627 -- F --> 2628
    2614 -- F --> 2617[Transfer: addresses of all replacements, their tense code and/or plural/singular flag from their Source-Phrase to the SDS; Store First-Elliptical-Phrase in each replacement phrase;]
    2614 -- T --> 2615[Current-Phrase = next consecutive phrase after Current-Phrase; Source-Phrase = next consecutive phrase after Source-Phrase;]
    2615 --> 2616{If Source-Phrase has a same phrase Current-Match with Current-Phrase}
    2616 -- T --> 2614
    2616 -- F --> 2637[Go to RETURN;]
    2617 --> 2618[RESTART = 2635; Store in the SDS: RESTART and its value, RETURN, Current-Match, Current-Sentence, TPSV;]
    2618 --> 2619{If ellipsis type has a special function}
    2619 -- T --> 2621[Evaluate the special function;]
    2619 -- F --> 2601[Go to 2601;]
    2621 --> 2601

```

FIG. 13B

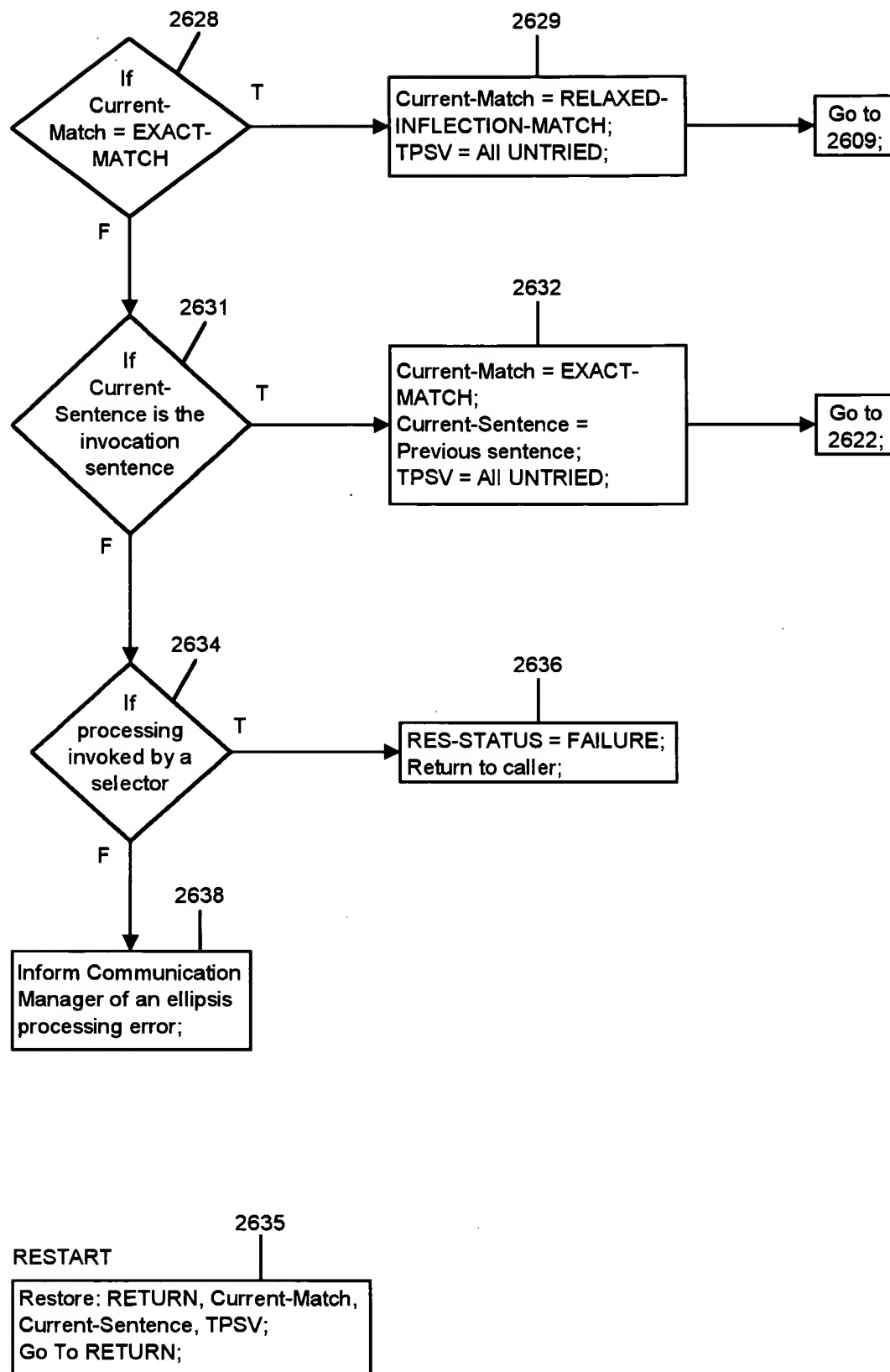


FIG. 13C

CONDITION	SUBJECT SOURCE
1. nonfinite verb or morphological word@ premodifies a concrete noun	premodified noun's state representation data structure is searched for the nonfinite verb or the morphological word@ in a clause relation
2. nonfinite verb or morphological word@ premodifies a noun	premodified noun
3. nonfinite verb or morphological word@ premodifies a state abstract noun	owner of the state abstract noun
4. nonfinite verb, verbless clause, or morphological word@ postmodifies a noun	postmodified noun
5. nonfinite verb, verbless clause, or morphological word@ postmodifies a state abstract noun	owner of the state abstract noun
6. a prepositional phrase modifies the verbal in the nonfinite verb clause or modifies the morphological word@	the complement of the modifying prepositional phrase
7. none	the main clause
8. nonfinite verb, verbless clause, or morphological word@ is not the subject of the clause	subject of the main clause
9. subject of the main clause is not first person pronoun	1st person (speaker)
10. subject of the main clause is not a second person pronoun, or not a direct address sentence	2nd person (person spoken to)
11. none	another untried noun in the main clause
12. none	the context
13. none	general reference (an indefinite pronoun)

FIG. 15A

CONDITION	OBJECT SOURCE
1. nonfinite verb or morphological word@ premodifies a noun	premodified noun
2. nonfinite verb or morphological word@ postmodifies a noun	postmodified noun
3. a prepositional phrase modifies the verbal in the nonfinite verb clause or modifies the morphological word@	the complement of the modifying prepositional phrase
4. 2nd person is not currently the subject source	2nd person (person spoken to)
5. 1st person is not currently the subject source	1st person (speaker)
6. none	another untried noun in the main clause
7. none	the context
8. none	default (most common object)
9. none	general reference (an indefinite pronoun)

FIG. 15B

```

graph TD
    26100{26100  
If  
ellipted  
clause is a  
premodifying nonfinite  
verb clause or a  
morphological  
word@}
    26101[26101  
Subject-Source-List = 1 thru 3,  
7 thru 13;  
Object-Source-List = 1, 4 thru 9;]
    26102{26102  
If  
ellipted  
clause is a  
postmodifying nonfinite  
verb clause or a  
morphological  
word@}
    26103[26103  
Subject-Source-List = 4 thru 13;  
Object-Source-List = 2 thru 9;]
    26104{26104  
If  
ellipted  
clause is a  
verbless  
clause}
    26105[26105  
Subject-Source-List = 4 thru 5,  
7 thru 13;]
    26106[26106  
Subject-Source-List = 6 thru 13,  
Object-Source-List = 3 thru 9;]
    26107{26107  
If  
subject  
is  
ellipted}
    26108[26108  
Create TSV;  
ESUB = True;  
SUBSHIFT = False;  
SUB = 1st value in  
Subject-Source-List;]
    26109{26109  
If  
object  
is  
ellipted}
    26110[26110  
Create TOV;  
EOBJ = True;  
OBJ = 1st value in  
Object-Source-List;]
    26111[26111  
Go to  
26111;]

    26100 -- T --> 26101
    26100 -- F --> 26102
    26101 --> 26107
    26102 -- T --> 26103
    26102 -- F --> 26104
    26103 --> 26107
    26104 -- T --> 26105
    26104 -- F --> 26106
    26105 --> 26107
    26106 --> 26107
    26107 -- T --> 26108
    26107 -- F --> 26109
    26108 --> 26109
    26109 -- T --> 26110
    26109 -- F --> 26111
    26110 --> 26111
  
```

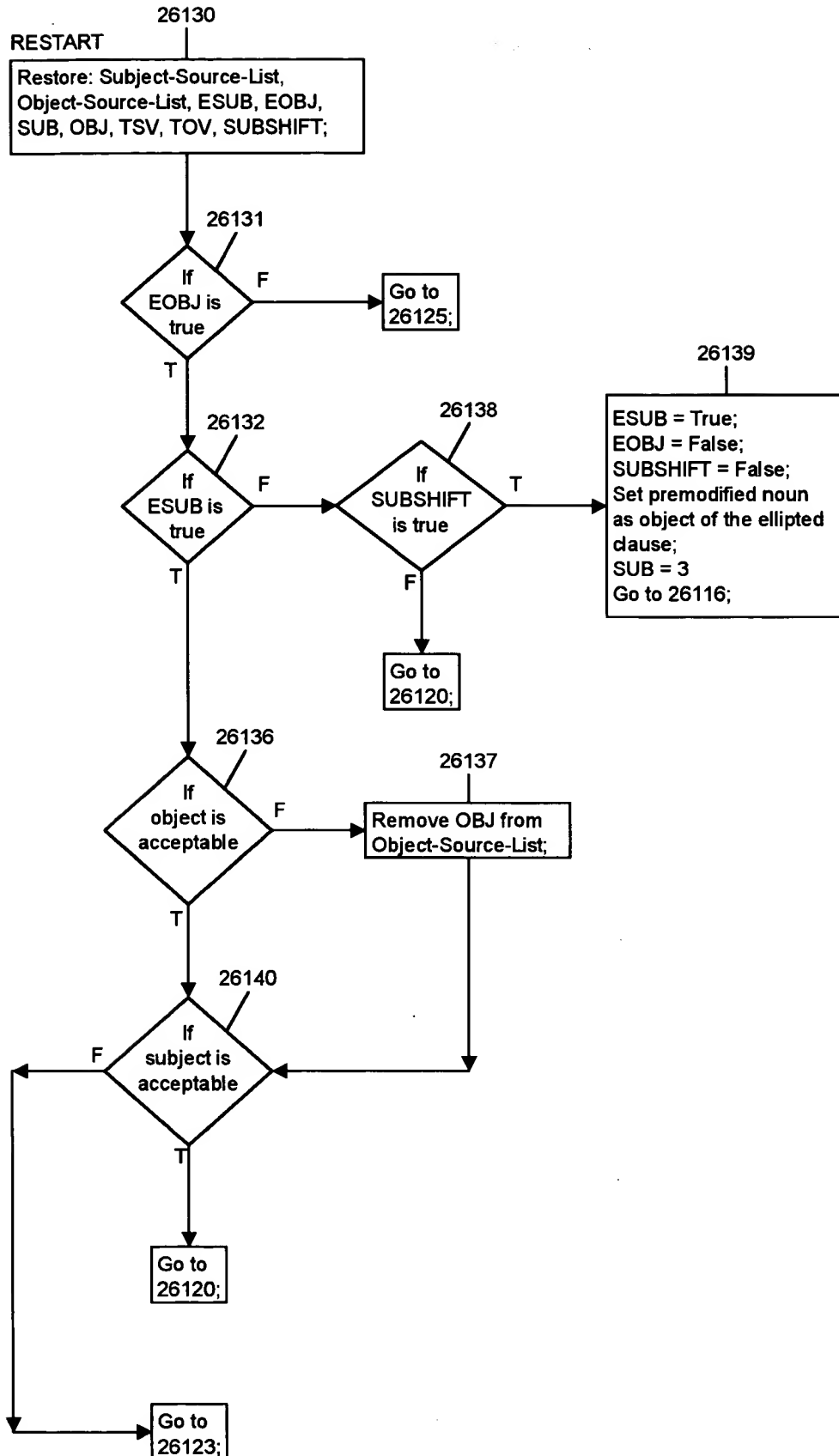



FIG. 16C

002260" 035T 2960

MEMORY 90 CONCRETE NOUN ENTRY FORMAT:
<u>Word Sense Number of Entry:</u>
<u>Set of States and Properties with their:</u> Word sense numbers and their addresses in Memory 80; Values or value ranges; Pointers to associated C-descriptors, S-descriptors, and T-descriptors;
<u>Partitioned Sets of Pointers to A-descriptors and T-descriptors:</u>
<u>Supertype Number:</u>
<u>Subtype Numbers:</u>

FIG. 17B

EXTERNAL RELATION STRUCTURE GENERAL FORMATS:	
MODIFIER INDEXED RELATIONS ENTRY FORMAT:	
<u>Word sense number(s) of a modifier:</u>	
<u>A-, C-, S-, or T- descriptor:</u>	
<u>Modifiee word sense entry or type number:</u>	
RELATION INDEXED FUNCTION A-RELATION ENTRY FORMAT:	
<u>Word sense number of verb in the function A-relation:</u>	
<u>A-descriptor:</u>	
<u>Modifiee word sense entry or type number:</u>	
RELATION INDEXED NON-FUNCTION RELATION ENTRY FORMAT:	
<u>A-, C-, S-, or T- descriptor:</u>	
<u>Modifiee word sense entry or type number:</u>	

FIG. 17C

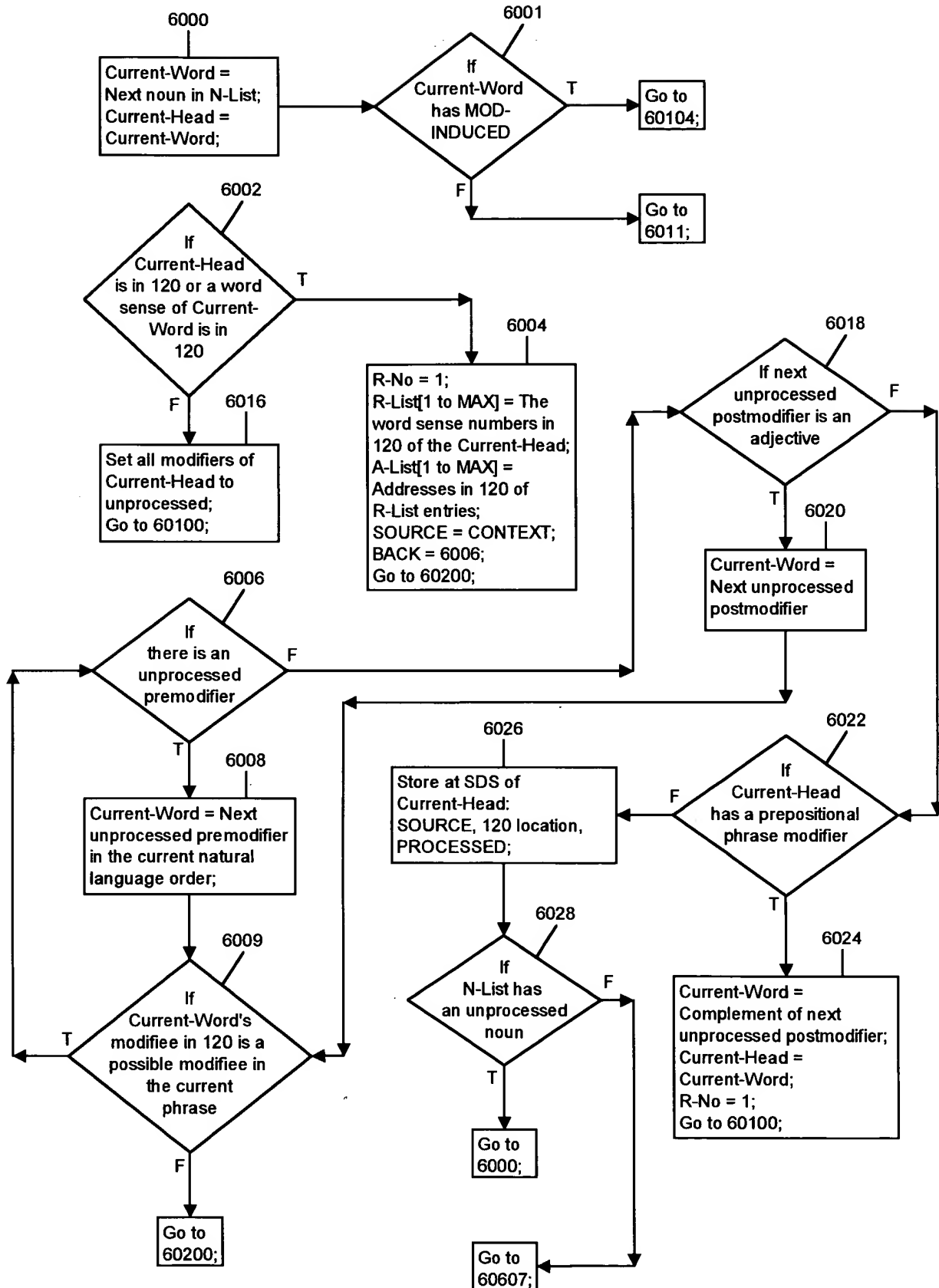


FIG. 17D

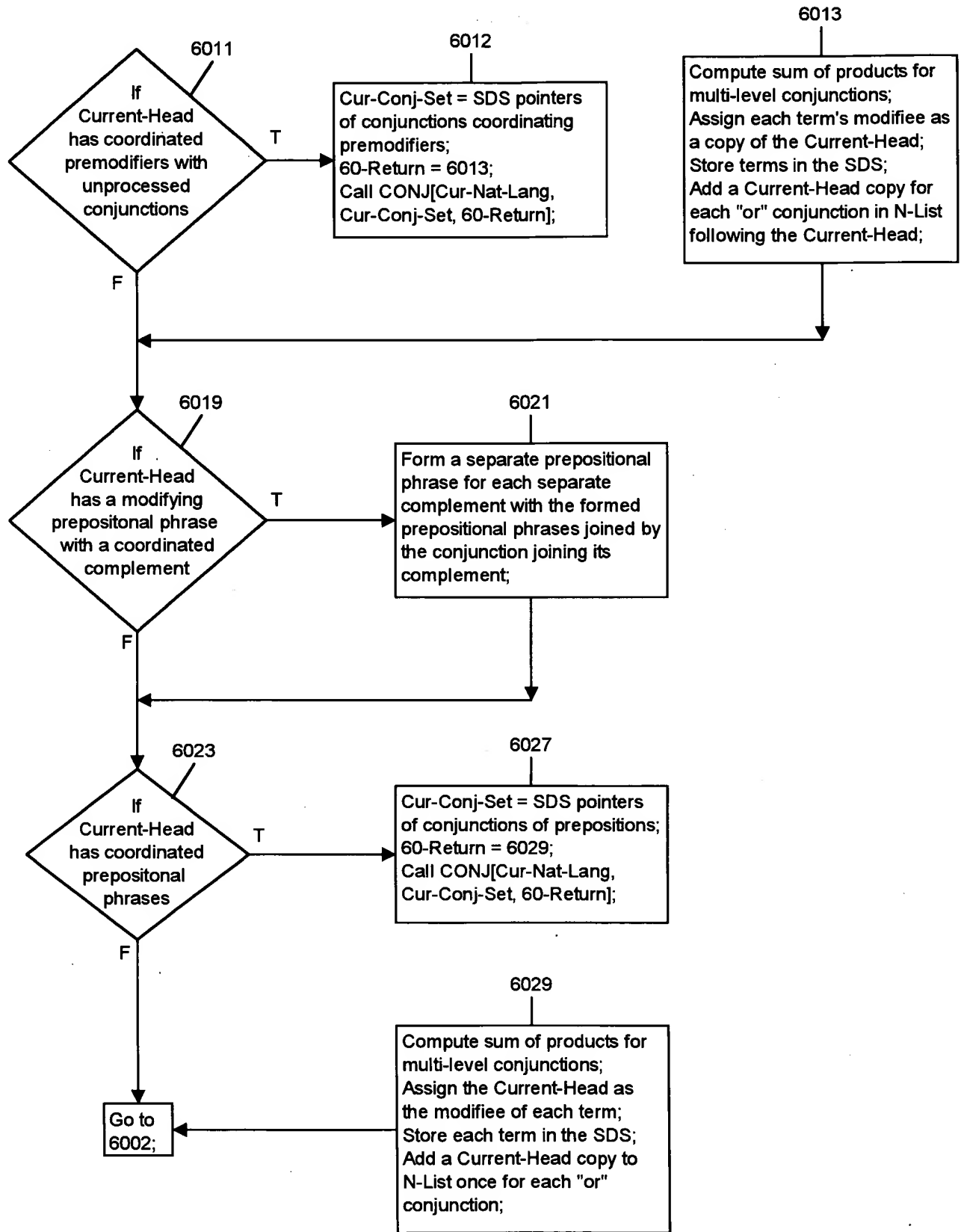


FIG. 17E

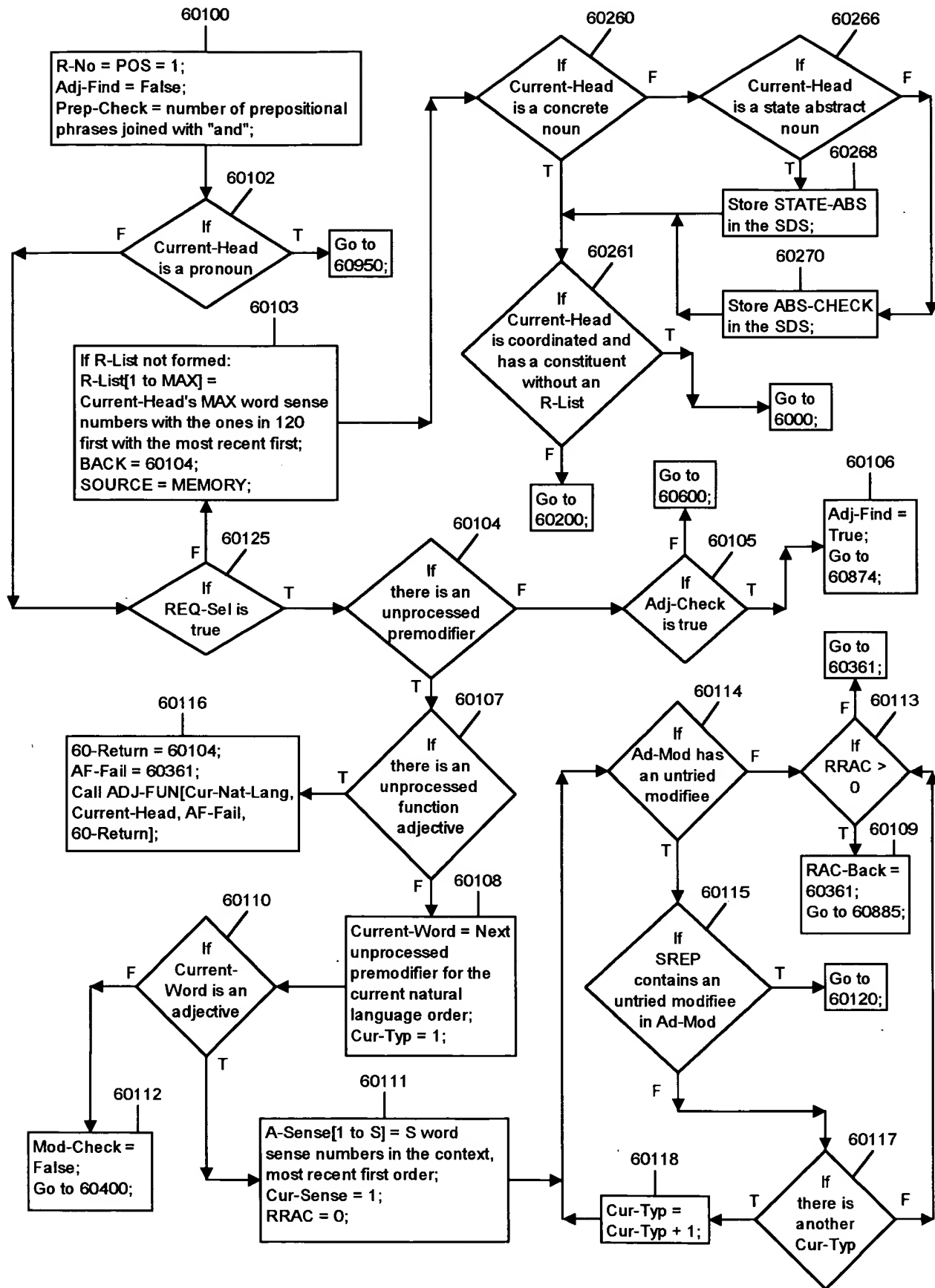


FIG. 17F

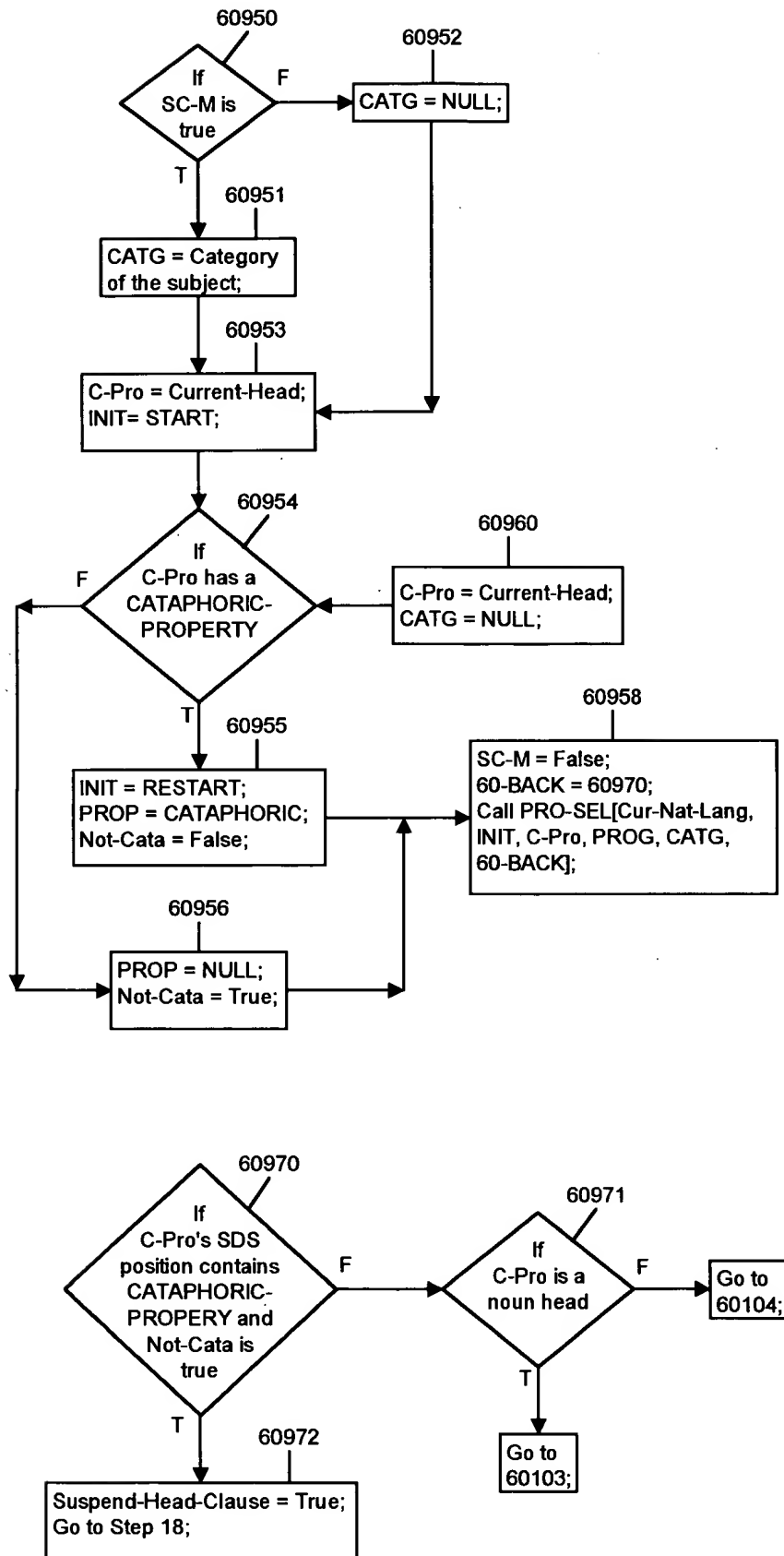


FIG. 17G

```

graph TD
    60200{If R-No < 0}
    60201{If Current-Head is an unprocessed subject, receiver, subject or object complement, or appositive}
    60203{If clause has a processed noun subject, "to be" verb, and an unprocessed noun or pronoun subject complement}
    60206{If clause has a processed noun subject, "to be" verb, and a processed noun subject complement}
    60211{If Current-Head is an unprocessed subject}
    60212{If clause verb is a pronoun without an R-List}
    60232{If SOURCE = CONTEXT and Current-Head is not a specific known reference or a pronoun}
    60208{If P-ADV is true}

    60200 -- T --> 60201
    60200 -- F --> 60232
    60201 -- T --> 60203
    60201 -- F --> 60208
    60203 -- T --> 60204[SC-M = True; Current-Head = Unprocessed subject complement; Go to 6000;]
    60203 -- F --> 60206
    60206 -- T --> 60280[Go to 60280;]
    60206 -- F --> 60211
    60211 -- T --> 60212
    60211 -- F --> 60215[Go to 60215;]
    60212 -- T --> 60940[Go to 60940;]
    60212 -- F --> 60214[Go to 60214;]
    60232 -- T --> 60234[R-No = MAX + 1; Go to 60103;]
    60232 -- F --> 60550[Go to 60550;]
    60208 -- T --> 60336[Go to 60336;]
    60208 -- F --> 60104[Go to 60104;]

```

FIG. 17H

```

graph TD
    60280{60280  
If  
there are  
unprocessed  
conjunctions}
    60281{60281  
If  
there is a  
respective  
function}
    60282[60282  
Form a clause for each subject  
composed of a subject, "to be"  
verb, and the corresponding  
subject complement;  
Join clauses with the subject  
conjunction;]
    60283{60283  
If  
there are  
coordinated  
subjects}
    60284[60284  
Cur-Conj-Set = All  
conjunctions joining subjects;  
60-Return = 60285;  
Call CONJ[Cur-Nat-Lang,  
Cur-Conj-Set, 60-Return];]
    60285[60285  
Form a clause for each coordinated  
noun phrase subject composed of a:  
noun phrase subject, "to be" verb,  
and the subject complement;  
Clauses are joined by conjunction(s)  
joining the noun phrase subject;]
    60286{60286  
If  
there are  
coordinated  
subject  
complements}
    60287[60287  
Cur-Conj-Set = All  
conjunctions joining subject  
complements;  
60-Return = 60288;  
Call CONJ[Cur-Nat-Lang,  
Cur-Conj-Set, 60-Return];]
    60288[60288  
For each stated or formed clause:  
form a separate clause for each  
group of subject complements joined  
by an "or", composed of the subject,  
"to be" verb, and group of subject  
complements;  
Join each formed clause with an "or";]
    60289{60289  
If one  
or more clauses  
have been  
formed}
    60290[60290  
Remove subject and subject complement  
noun phrases from N-List and SDS;  
Add an N-List and SDS entry for all noun  
phrases in all formed clauses;  
Add each formed clause to the SDS;]
    60291{60291  
If  
there is an  
unprocessed  
clause}
    60292[60292  
Current-Head = 1st unprocessed noun  
head in the next subject/subject  
complement clause;  
BACK = 60104;  
Go to 60104;]
    60293[60293  
Search the subject's R-List for a match  
with the R-List elements of the next  
unprocessed subject complement for  
ID# and type;]
    60294{60294  
If a  
match is  
found}
    60295[60295  
Set R-No to the match or to the value needed;  
For subject and subject complement:  
REQ = R-No for subject/subject complement;  
Current-Head = Subject;  
BACK = 60296;  
Go to 60392;]
    60296[60296  
Current-Head = Subject complement;  
BACK = 60296 or 60291 for the last subject  
complement;  
Go to 60392;]
    60297{60297  
If the  
SOURCE is  
CONTEXT}
    60298[60298  
Form an R-List as at 60103 for subject  
and/or subject complement with a  
CONTEXT SOURCE;]
    60299[60299  
Compose a new subject complement  
composed of the old subject  
complement modifying the subject;  
Set subject to PROCESSED;]

    60280 -- F --> 60290
    60280 -- T --> 60281
    60281 -- T --> 60282
    60281 -- F --> 60283
    60283 -- T --> 60284
    60283 -- F --> 60286
    60286 -- T --> 60287
    60286 -- F --> 60289
    60289 -- T --> 60290
    60289 -- F --> 60291
    60291 -- T --> 60293
    60291 -- F --> 60292
    60293 --> 60294
    60294 -- T --> 60295
    60294 -- F --> 60297
    60297 -- T --> 60298
    60297 -- F --> 60299
    60299 --> 60296
    60296 --> 60290

```

FIG. 17 I

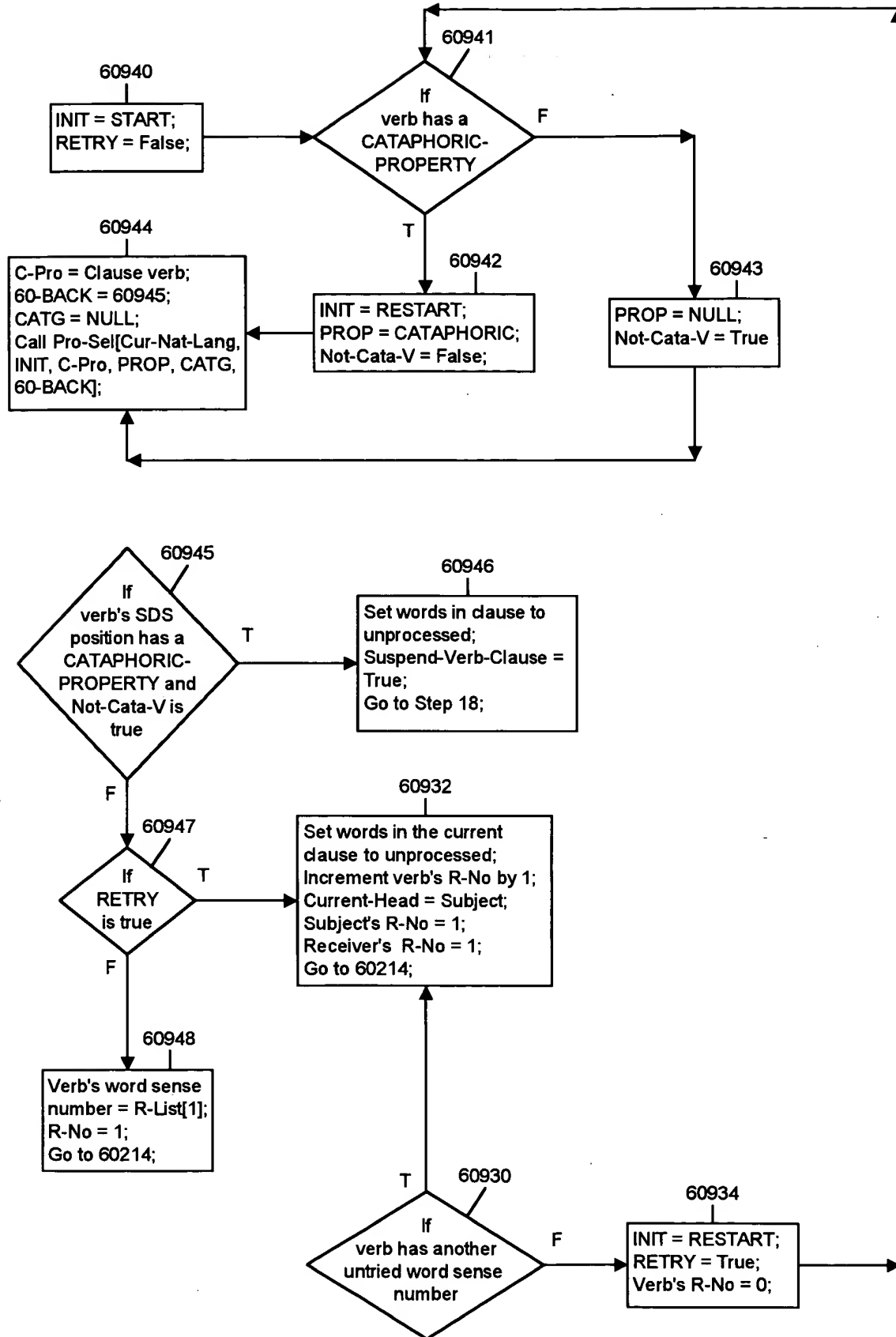


FIG. 17J

```

graph TD
    60214[70-SEL-D = Pointer to R-List of Current-Head;  
70-SEL-V = Pointer to verb's SDS position;  
70-SEL-R = NULL;  
70-TYP-SEL = DV;] --> 60217[60-Return = 60230;  
Call 70[70-TYP-SEL,  
70-SEL-D, 70-SEL-R,  
REQ, 60-Return];]
    60215{If Current-Head  
is an unprocessed object} -- T --> 60216[70-SEL-D = Selected word sense of the clause subject;  
70-SEL-V = Selected word sense of the clause verb;  
70-SEL-R = Pointer to the R-List of Current-Head;  
70-TYP-SEL = R;]
    60215 -- F --> 60218[Form a clause with the modifiee of the Current-Head as subject, with a "to be" verb, the Current-Head as subject complement;  
Store the clause in the SDS;]
    60218 --> 60203[Go to 60203;]
    60216 --> 60217
    60230{If 70-TYP-SEL = DV} -- T --> 60240[Subject's R-No = 70-SEL-D;  
Receiver's noun phrase = unprocessed;  
Store 70-SEL-V in the SDS;  
Receiver's R-No(s) = 1;]
    60230 -- F --> 60246{If subject's word sense is changed}
    60246 -- F --> 60244[Receiver's R-No(s) = 70-SEL-R;  
Store 70-SEL-V in the SDS;]
    60246 -- T --> 60248[Subject's R-No = 70-SEL-D;  
Subject's noun phrase = unprocessed;  
BACK = 60244;  
Current-Word = Current-Head;]
    60240 --> 60242[BACK = 60104;  
Current-Word = Current-Head;]
    60244 --> 60242
    60242 --> 60394[Current-Word-Status = PROCESSED;  
Store at Current-Word's SDS position: R-No, MAX, R-List, Current-Word-Status, SOURCE, R-RAC, RRAC;  
SREP[POS, 1] = R-List[R-No];  
Go to BACK;]
    60248 --> 60392[SREP[POS, 2] = Cur-Typ;  
SREP[POS, 3] = Position of Current-Modifiee or NULL for a noun head;  
SREP[POS, 4] = Modifiee's location of modifier relation or REQ for a noun head;]
    60392 --> 60394

```

FIG. 17K

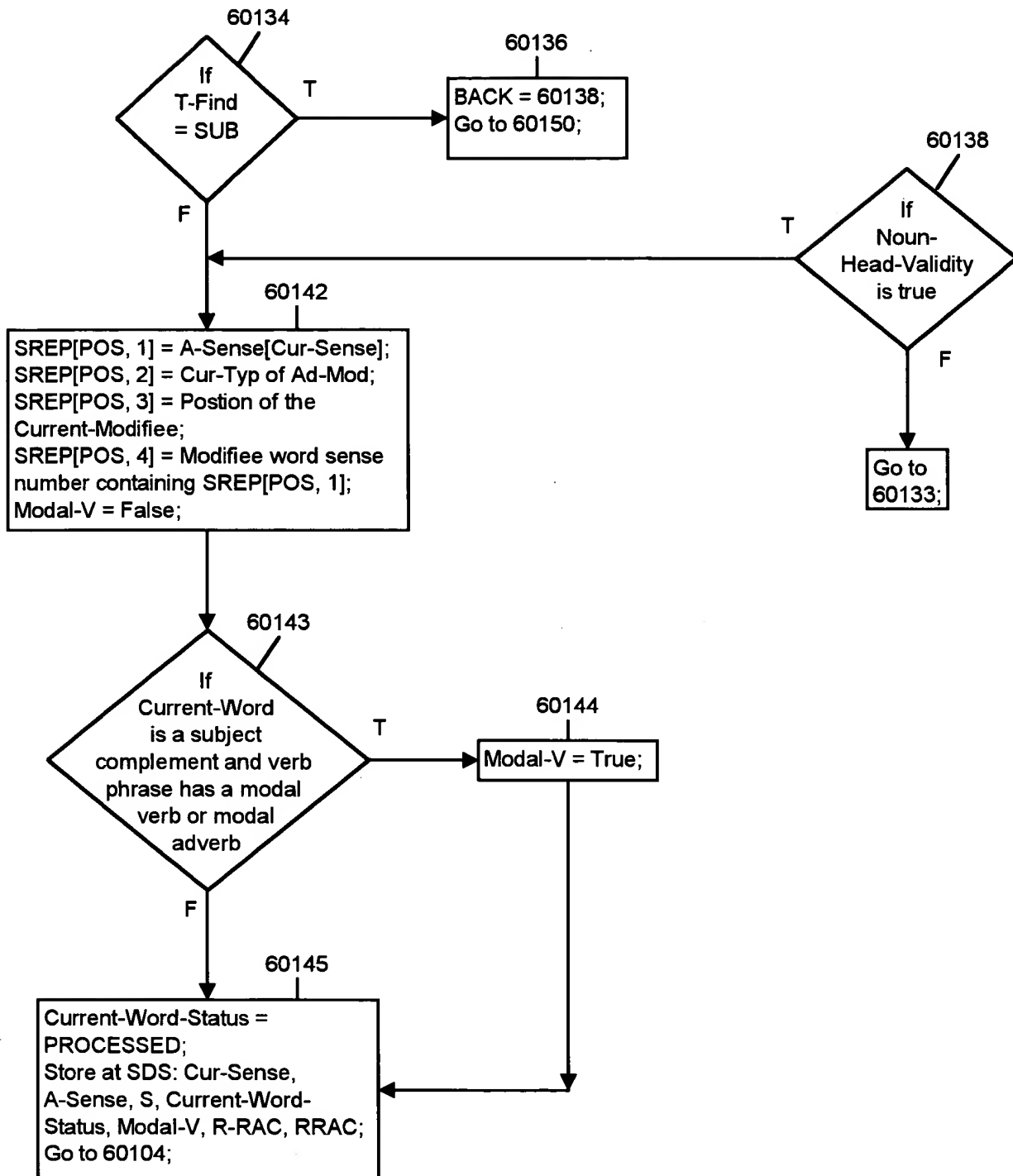


FIG. 17M

```

graph TD
    60150["Noun-Head-Validity = True;  
1-Mod = False;  
TEMP = Current-Modifiee;  
Set type of TEMP implied  
by Current-Word;  
3-Mod = False;"] --> 60151{"If  
processed  
modifiers of  
Current-Modifiee  
have a modification  
relation with  
TEMP"}
    60151 -- F --> 60158a["Go to  
60158;"]
    60151 -- T --> 60152{"If  
Current-Head  
= Current-  
Modifiee"}
    60152 -- F --> 60162{"If  
Current-  
Modifiee modifies the  
Current-Head"}
    60152 -- T --> 60154["TEMP-H = TEMP;  
1-Mod = True;"]
    60154 --> 60156{"If  
TEMP-H has  
all stated state and  
property values, and REQ  
and verb requiements  
are not violated"}
    60156 -- F --> 60158b["Noun-Head-Validity  
= False;"]
    60158b --> 60160["Go to BACK;"]
    60156 -- T --> 60177["Go to  
60177;"]
    60162 -- T --> 60164{"If  
TEMP sets  
a type for the  
Current-Head"}
    60162 -- F --> 60170["TEMP-SH = Modifiee  
of Current-Modifiee;  
3-Mod = True;"]
    60164 -- T --> 60166["TEMP-H =  
Current-Head;  
Set type of  
TEMP-H;"]
    60164 -- F --> 60172{"If  
TEMP  
sets a type for  
TEMP-SH"}
    60166 --> 60168{"If  
TEMP has a  
modification  
relation with  
Current-Head"}
    60168 -- T --> 60169{"If  
TEMP has  
a new  
modification  
relation"}
    60168 -- F --> 60158b
    60169 -- T --> 60171["Current-Modifiee's  
SREP[POS, 4] = The  
new modification  
relation;"]
    60171 --> 60177
    60169 -- F --> 60175{"If  
processed  
modifiers of modifiee  
of Current-Modifiee  
have a modification  
relation with  
TEMP-SH"}
    60175 -- F --> 60158a
    60175 -- T --> 60176{"If  
TEMP has  
a related  
modification  
relation with  
TEMP-SH"}
    60176 -- T --> 60169
    60176 -- F --> 60158a
    60172 -- T --> 60174["Set type of  
TEMP-SH;  
TEMP = TEMP-SH;"]
    60174 --> 60175
    60170 --> 60172

```

FIG 17N

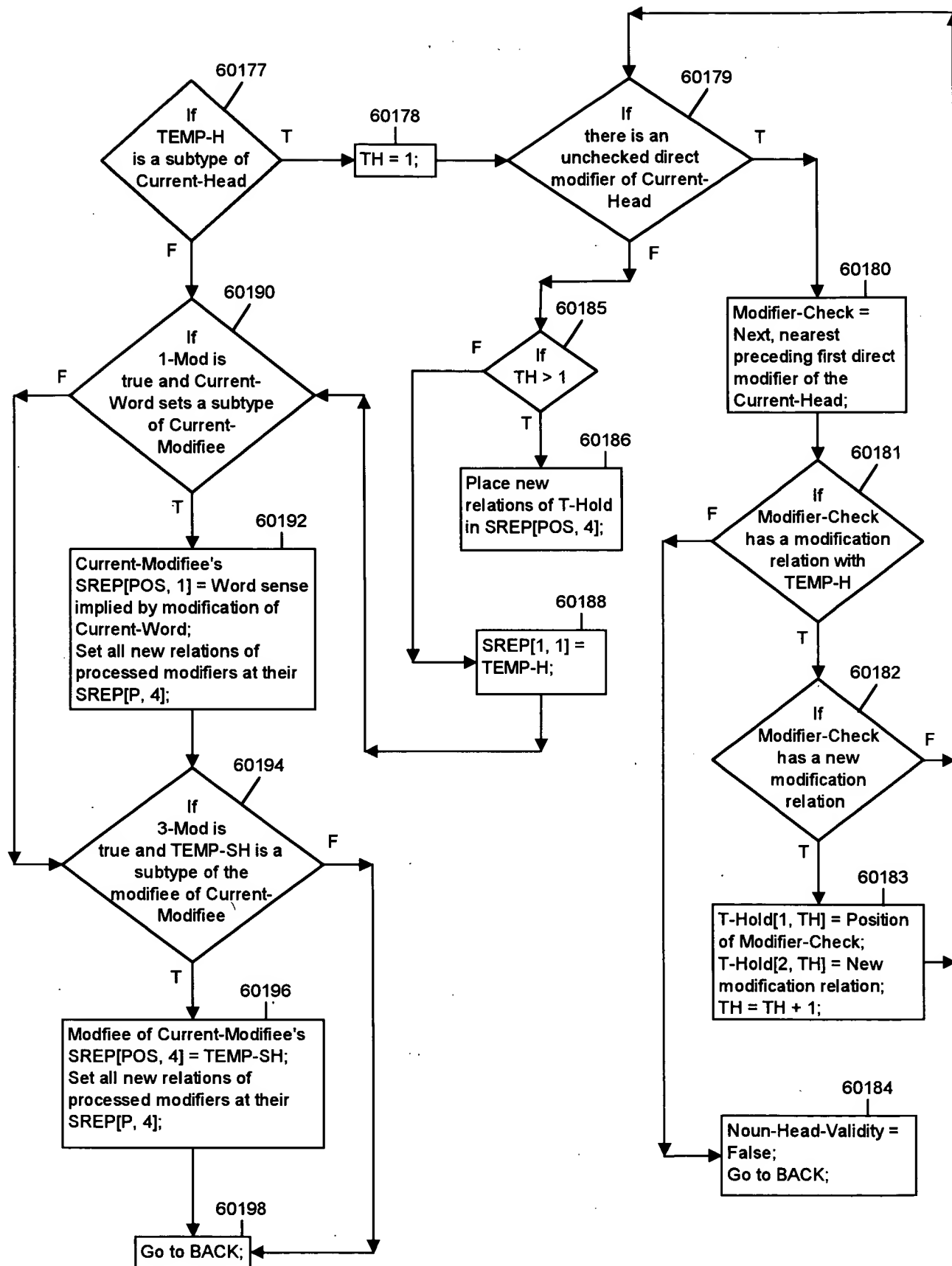


FIG. 170

```

graph TD
    60400[60400  
R-List[1 to MAX] = MAX word senses of Current-Word in the order in 120 and most recent first;  
R-No = Cur-Typ + 1;  
RRAC = 0;] --> 60402[60402  
Current-Modifiee = Next possible modifiee in N-Mod[Current-Word, Cur-Typ, Cur-Nat-Lang];  
Sep-Check = The number of modifiers which precede the Current-Word;]
    60402 --> 60406[60406  
TRIED = TMV[Current-Word WS#, Current-Modifiee's position, Current-Modifiee WS#];  
SUCCEED = SMV[Current-Word WS#, Current-Modifiee's position, Current-Modifiee WS#]]
    60406 --> 60408{60408  
If TRIED = true}
    60408 -- F --> 60410[60410  
TMV[Current-Word WS#, Current-Modifiee's position, Current-Modifiee WS#] = True;]  
60410 --> 60418[60418  
Search for a non-conflicting modifier indexed relation in the possible partitions of Current-Modifiee's external relation structure for a super-type, match or subtype;]  
60418 --> 60419{60419  
If a relation is found}
    60419 -- T --> 60424[60424  
SMV[Current-Word WS#, Current-Modifiee's position, Current-Modifiee WS#] = True;]  
60424 --> 60430{60430  
If Noun-Head-Validity is true;}
    60430 -- T --> 60429[60429  
BACK = 60104;  
Go to 60732;]  
60430 -- F --> 60426{60426  
If R-List[R-No] implies a subtype}
    60426 -- T --> 60428[60428  
BACK = 60430;  
Go to 60150;]  
60426 -- F --> 60414{60414  
If there is another possible modifier}
    60414 -- T --> 60402
    60414 -- F --> 60415{60415  
If R-No < MAX}
    60415 -- T --> 60416[60416  
R-No = R-No + 1;]  
60416 --> 60417{60417  
If there is another Cur-Typ}
    60417 -- T --> 60422[60422  
R-No = 1;  
Cur-Typ = Cur-Typ + 1;]  
60422 --> 60400
    60417 -- F --> 60434{60434  
If RRAC < 0}
    60434 -- T --> 60446[60446  
RAC-Back = 60436;  
Go to 60885;]  
60434 -- F --> 60436[60436  
Go to 60436;]  
60415 -- F --> 60421{60421  
If there is a relation with an alternative type or conflicting value}
    60421 -- T --> 60423[60423  
RRAC = RRAC + 1;  
R-RAC[RRAC, 1] = word in conflict;  
R-RAC[RRAC, 2] = Current-Modifiee;  
R-RAC[RRAC, 3] = Sep-Check;]  
60423 --> 60421
    60421 -- F --> 60414

```

FIG. 17P

```

graph TD
    60885[60885  
CONF = 0;  
F-R = -1;] --> 60886{60886  
If  
there is an  
untried row in  
R-RAC}
    60886 -- T --> 60887[60887  
C-R = Next untried  
row # in R-Rac;]
    60886 -- F --> 60893{60893  
If  
CONF > 0}
    60887 --> 60889{60889  
If  
R-RAC[C-R, 1]  
modifies Current-  
Modifiee, is the same type  
as Current-Word, and  
R-RAC[C-R, 3] > 0}
    60889 -- T --> 60890{60890  
If  
F-R < 0}
    60889 -- F --> 60891[60891  
F-R = C-R;]
    60890 -- T --> 60891
    60890 -- F --> 60886
    60891 --> 60886
    60893 -- T --> 60894[60894  
Order rows in CONF-M by  
CONF-M[x, 1] nearest in  
position to the Current-Word;  
Store at SDS: CONF,  
CONF-M;]
    60893 -- F --> 60888{60888  
If  
F-R > 0}
    60894 --> 60888
    60888 -- T --> 60896[60896  
Create a copy of Current-Word's  
noun phrase minus R-RAC[F-R,  
1] in the SDS;  
Mark head with MOD-INDUCED  
in the SDS;  
Remove the Current-Word and  
its modifiers from the noun  
phrase;  
Store the head of the copied  
phrase after the head of the  
current phrase;  
Join copied phrase with an "and"  
conjunction to the sentence;]
    60888 -- F --> 60895[60895  
Go to  
RAC-Back;]
    60896 --> 60897{60897  
If  
Current-Word  
is a prepositional  
complement  
head}
    60897 -- T --> 60603[Go to  
60603;]
    60897 -- F --> 60104[Go to  
60104;]

```

FIG. 17Q

```

graph TD
    60436{If Current-Word is a verbal morphological word@}
    60427[Call ellipsis processing at RESTART in the Current-Word's SDS position;]
    60437[Cur-Rel-Set = All A-relations except function A-Relations without A-Relation sentence roles;  
RRAC = 0;  
60-Back = 60438;  
Cur-Typ = 1;  
R-No = R-No + 1;]
    60448{If Cur-Rel-Set is empty}
    60443{If Current-Word is PROCESSED}
    60435[R-No = 1;]
    60444[Current-Modifiee = Next possible modifiee in N-Mod[Current-Word, Cur-Typ, Cur-Nat-Lang];  
Sep-Check = Number of modifiers of the Current-Modifiee which precede the Current-Word;  
Go to 60450;]
    60440{If there is a possible modifiee}
    60441{If R-No < MAX}
    60442[R-No = R-No + 1;]
    60443_2{If there is another Cur-Typ}
    60449[Cur-Typ = Cur-Typ + 1;  
R-No = 1;]
    60445{If RRAC > 0}
    60447[RAC-Back = 60355  
Go to 60355;]
    60438{If Current-Relation-Found is true}
    60439[BACK = 60104;  
Go to 60732;]

    60436 -- T --> 60427
    60436 -- F --> 60437
    60437 --> 60448
    60448 -- T --> 60445
    60448 -- F --> 60443
    60443 -- T --> 60443_2
    60443 -- F --> 60435
    60435 --> 60440
    60440 -- T --> 60440
    60440 -- F --> 60441
    60441 -- T --> 60442
    60441 -- F --> 60443_2
    60443_2 -- T --> 60449
    60443_2 -- F --> 60445
    60445 -- T --> 60447
    60445 -- F --> 60443_2
    60447 --> 60440
    60449 --> 60440
    60442 --> 60440
    60440 --> 60444
    60444 --> 60440
    60438 -- T --> 60439
    60438 -- F --> 60440
    60439 --> 60440
    60440 --> 60440

```

FIG. 17R

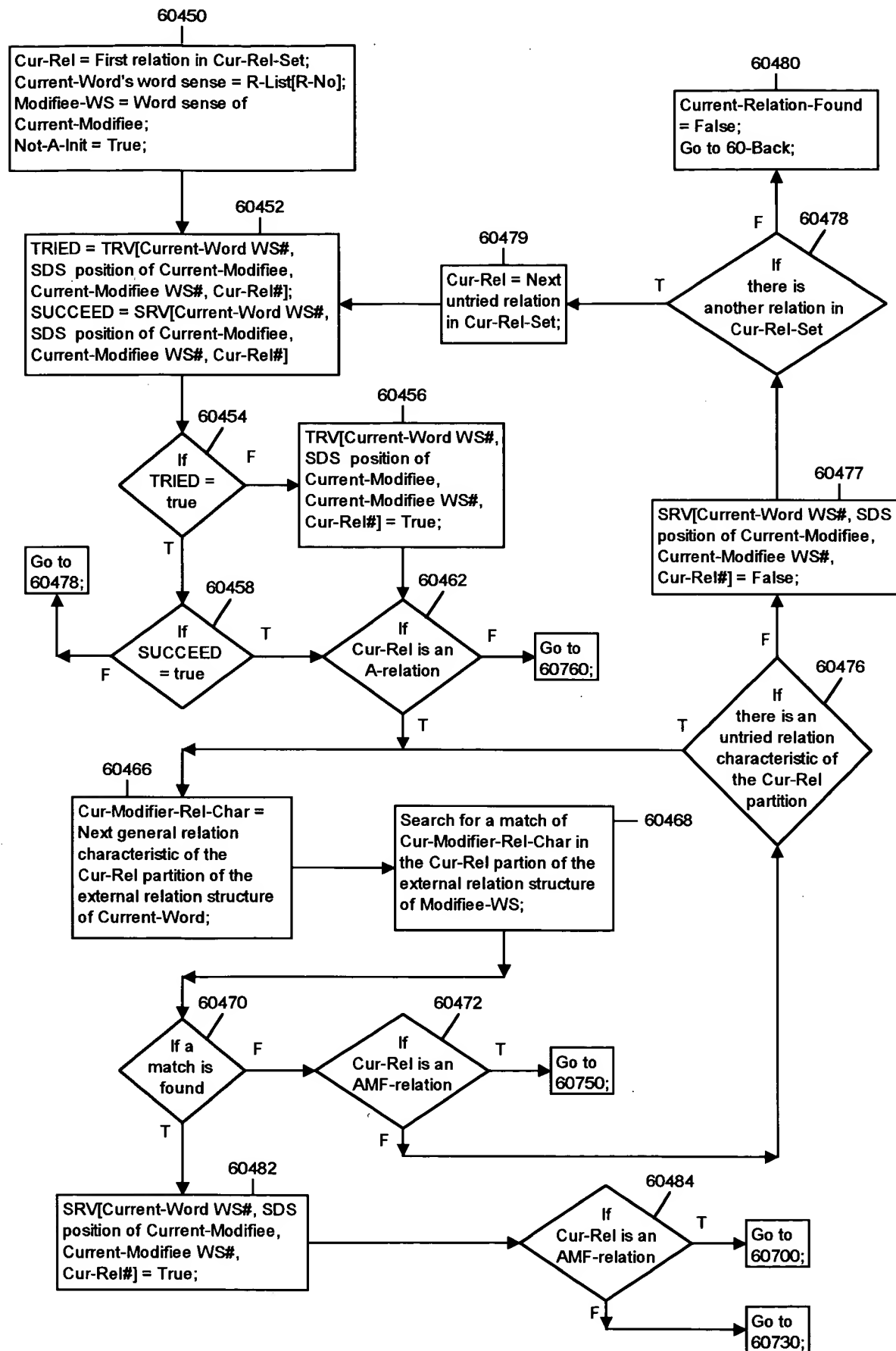


FIG. 17S

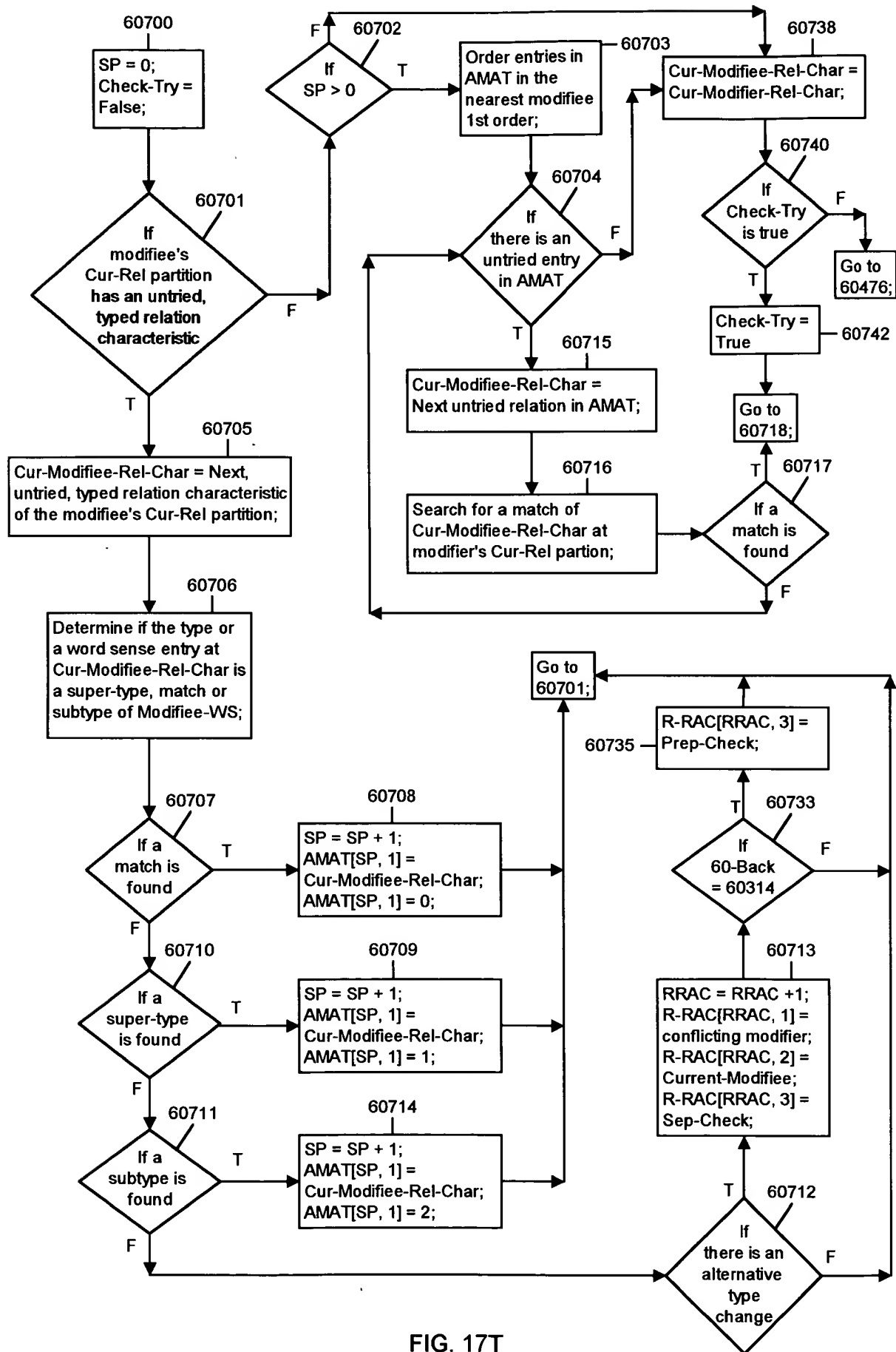


FIG. 17T

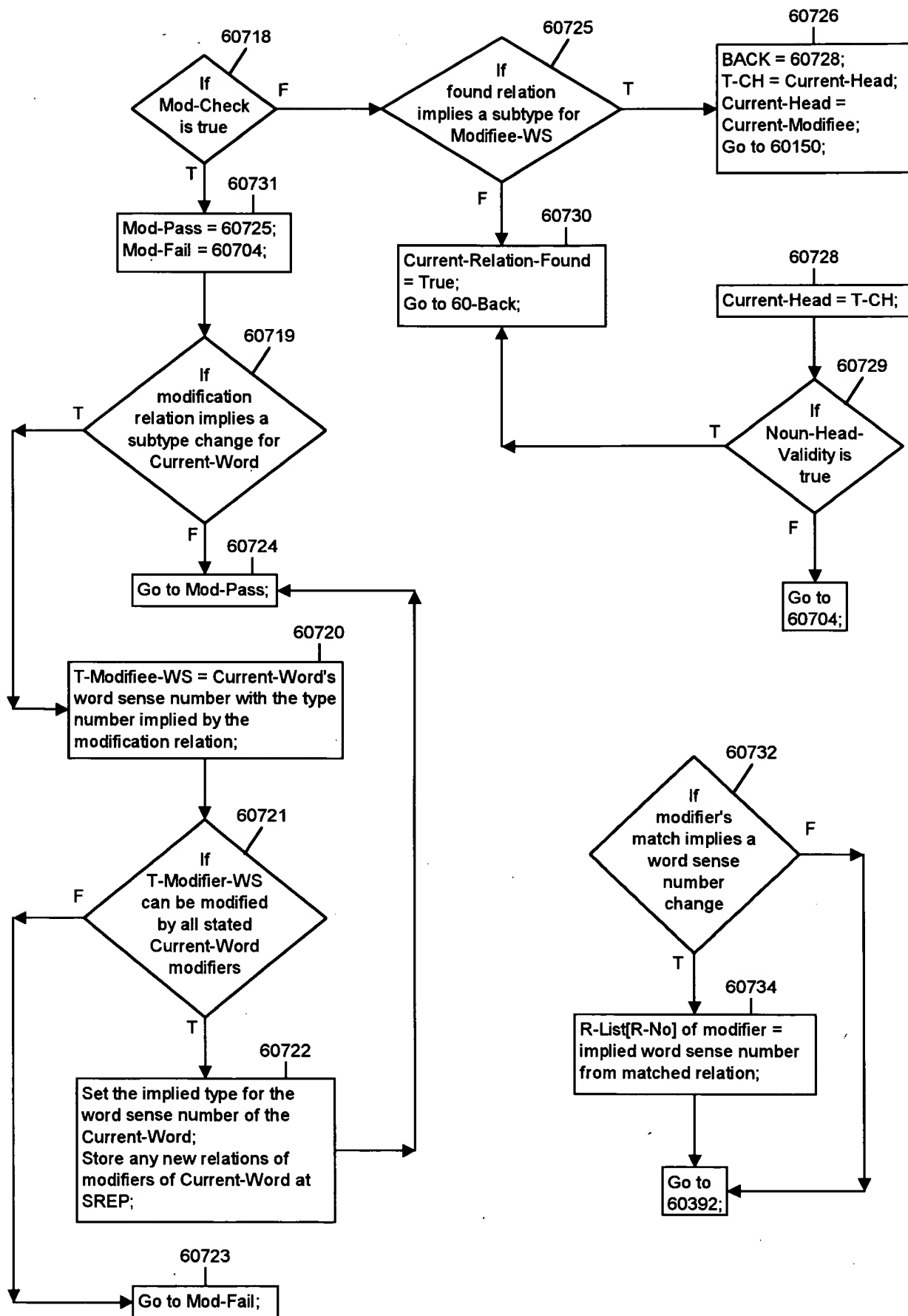


FIG. 17U

```

graph TD
    60750{60750  
If  
function  
A-relation has an  
A-relation for a  
sentence role of  
the clause  
relation}
    60752[60752  
Go to  
60476;]
    60754[60754  
Search for Modifier-WS  
in the sentence role  
A-relation(s);]
    60756{60756  
If  
Modifier-WS  
is found}
    60730[60730  
Go to  
60730;]

    60750 -- F --> 60752
    60750 -- T --> 60754
    60754 --> 60756
    60756 -- F --> 60752
    60756 -- T --> 60730
  
```

FIG. 17V

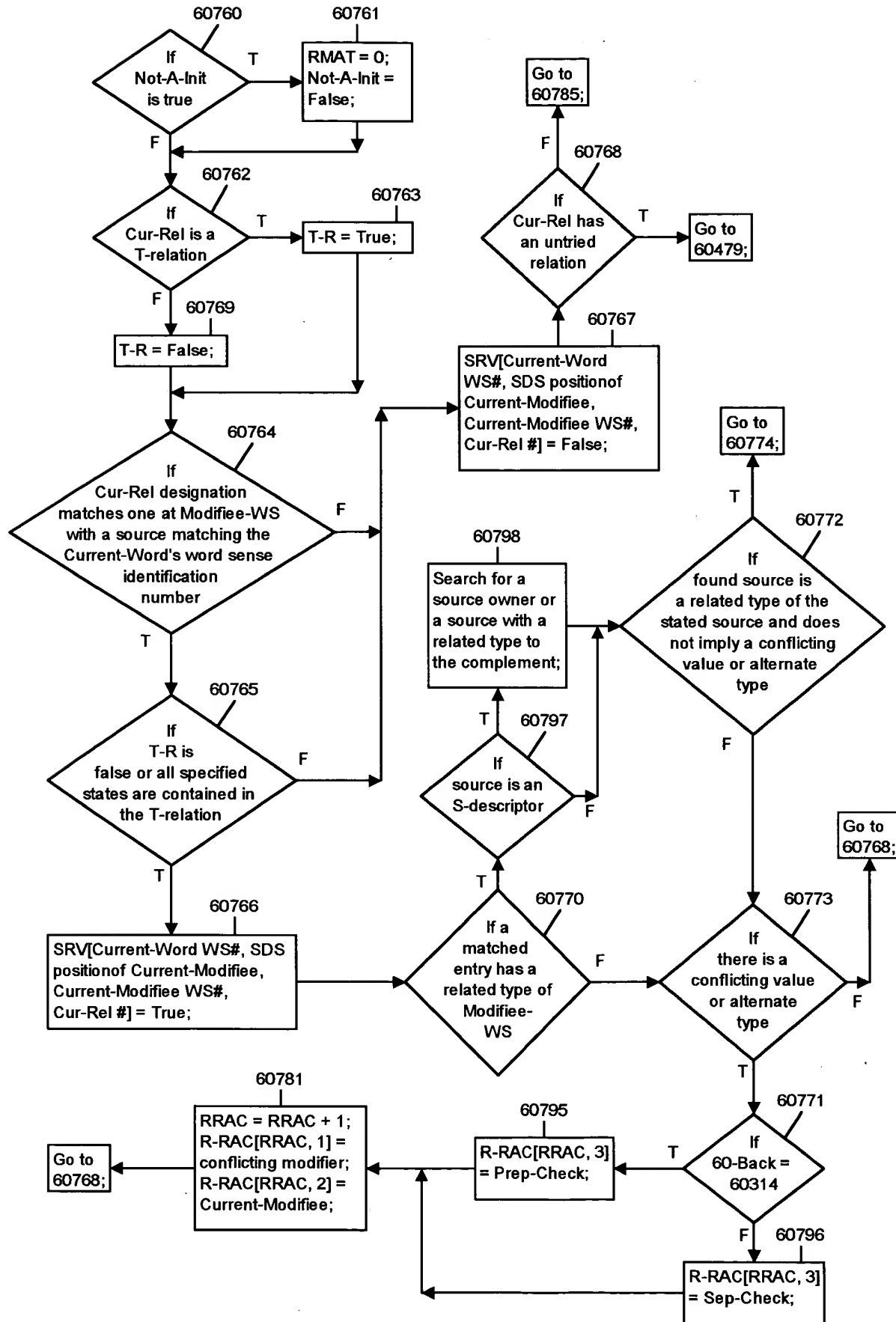


FIG. 17W

FIG. 17X

```

graph TD
    60600{60600  
If next  
unprocessed word is  
a postmodifying  
adjective}
    60601[60601  
Current-Word =  
Postmodifying adjective;  
Cur-Typ =  
@Postmodifying-Adjective;  
Go to 60120;]
    60602[60602  
Mark and store SREP  
elements for each  
modifier and noun at  
their SDS positions;]
    60603{60603  
If next  
unprocessed  
postmodifier is a  
prepositional  
phrase}
    60604{60604  
If  
prepositional  
phrase has an  
unprocessed  
preposition}
    60605[60605  
Cur-Typ = 1;  
Go to 60300;]
    60606[60606  
Current-Head = Head of  
prepositional phrase  
complement;  
Go to 60100;]
    60607{60607  
If  
current clause  
has a processed  
subject, a "to be" verb,  
an unprocessed  
adjective complement  
modified by a  
prepositional  
phrase}
    60608{60608  
If  
adjective is  
modified by an  
unprocessed  
prepositional  
phrase  
complement}
    60609[60609  
Go to  
Step 18;]
    60610{60610  
If  
current clause  
has a processed  
subject, a "to be" verb,  
and an unprocessed  
subject  
complement}
    60611{60611  
If  
subject  
complement is  
an adjective}
    60612[60612  
Current-Word = Adjective;  
Cur-Typ = @Subject;  
Go to 60120]
    60613{60613  
If  
subject  
complement is a  
prepositional  
phrase}
    60614{60614  
If  
preposition is  
unprocessed}
    60615[60615  
Current-Head =  
Head of  
prepositional  
complement;  
Cur-Typ = @Subject;  
Go to 60300;]
    60616{60616  
If  
subject  
complement is  
a noun or  
pronoun}
    60617{60617  
If next  
head has Abs-  
Check in its SDS  
position}
    60618{60618  
If  
N-List has an  
unprocessed  
noun}
    60619[60619  
Go to  
60900;]
    60620[60620  
Go to 6000;]
    60621[60621  
Go to 607002;]

    60600 -- T --> 60601
    60600 -- F --> 60602
    60602 --> 60603
    60603 -- T --> 60604
    60603 -- F --> 60607
    60604 -- T --> 60605
    60604 -- F --> 60606
    60605 --> 60603
    60606 --> 60608
    60607 -- T --> 60611
    60607 -- F --> 60610
    60608 -- F --> 60609
    60608 -- T --> 60606
    60609 --> 60603
    60610 -- T --> 60611
    60610 -- F --> 60618
    60611 -- T --> 60612
    60611 -- F --> 60613
    60612 --> 60610
    60613 -- T --> 60614
    60613 -- F --> 60616
    60614 -- T --> 60615
    60614 -- F --> 60606
    60615 --> 60613
    60616 -- T --> 60619
    60616 -- F --> 60617
    60617 -- T --> 60621
    60617 -- F --> 60618
    60618 -- T --> 60620
    60618 -- F --> 60610
  
```

FIG.17Y


```

graph TD
    60912[60912  
Set Best-Def as the  
modifiee of Next-Def's  
modifiers in the SDS;  
Store Modal-V at each  
transferred modifier;] --> 60914{60914  
If  
C-R is false, and  
Best-Def and Next-Def  
have different type  
numbers}
    60914 -- T --> 60915[60915  
Set Next-Def as having a  
TYPE-DEFINING  
relationship to Best-Def;]
    60914 -- F --> 60918{60918  
If  
there is an  
unprocessed  
Next-Def}
    60915 --> 60918
    60918 -- T --> 60919[60919  
Select Best-Def  
and Next-Def;  
C-R = False;  
Go to 60905]
    60918 -- F --> 60920{60920  
If  
Form-C is  
true}
    60920 -- T --> 60924{60924  
If  
there is one  
unprocessed  
formed clause}
    60920 -- F --> 60921{60921  
If there  
is a clause  
with a subject  
complement and an  
unprocessed sentence  
role word sense  
number}
    60924 -- T --> 60925[60925  
Form-C = False;]
    60924 -- F --> 60926[60926  
Select Best-Def and Next-Def,  
or the next formed clause;  
C-R = False;  
Go to 60905;]
    60925 --> 60926
    60921 -- T --> 60922[60922  
Current-Head = Next  
unprocessed sentence  
role of such a clause;  
BACK = 60104;  
Go to 60104;]
    60921 -- F --> 60923[60923  
Modal-V = False;  
Go to 60618;]
    
```

FIG. 17AA

```

graph TD
    60850[60850  
Cur-Prep = Preposition  
modifying the adjective;  
SUBJ = Word sense number  
of clause subject;  
COMP = Complement of the  
prepositional phrase;  
ADJ = The adjective;  
Return-60 = 60854;  
Modal-V = False;  
P-Call = False;] --> 60851{60851  
If  
verb phrase  
has a modal or  
modal  
adverb}
    60851 -- T --> 60852[60852  
Modal-V  
= True;]
    60851 -- F --> 60853
    60852 --> 60853
    60853[60853  
Call ADJ-PREP[Cur-Nat-Lang,  
SUBJ, ADJ, Cur-Prep, COMP,  
Return-60, Adj-Prep-Status,  
Current-Relation];] --> 60854{60854  
If  
Adj-Prep-Status  
= SEARCH}
    60854 -- T --> 60857
    60854 -- F --> 60855
    60857[60857  
60-Back = 60858;  
Cur-Rel-Set = Current-Relation;  
Current-Modifiee = SUBJ;  
Mod-Check = True;  
Go to 60450;] --> 60858{60858  
If  
Current-  
Relation-Found  
is true}
    60858 -- T --> 60860
    60858 -- F --> 60864
    60860[60860  
Current-Relation =  
Address of found relation;  
Adj-Prep-Status =  
FOUND-IN-90;] --> 60870
    60870[60870  
Return ADJ-PREP[  
Cur-Nat-Lang, SUBJ, ADJ,  
Cur-Prep,COMP, Return-60,  
Adj-Prep-Status,  
Current-Relation];] --> 60853
    60864[60864  
Current-Relation =  
NULL;  
Adj-Prep-Status =  
NOT-IN-90;] --> 60853
    60855{60855  
If  
Adj-Prep-  
Status =  
ADJ-FIND} -- T --> 60872
    60855 -- F --> 60856
    60856{60856  
If  
Adj-Prep-  
Status =  
FAIL} -- F --> 60861
    60856 -- T --> 60859
    60859{60859  
If  
P-Call  
is true} -- F --> 60868
    60859 -- T --> 60867
    60867[60867  
Return to  
caller;] --> 60853
    60868[60868  
Current-Word  
= ADJ;  
Go to 60360;] --> 60861
    60861{60861  
If  
Current-  
Relation is an  
AMF- or T-  
relation} -- T --> 60865
    60861 -- F --> 60618
    60865[60865  
BACK =  
60618;] --> 60390
    60390[60390  
SREP[POS, 2] =  
Cur-Typ;  
SREP[POS, 2] =  
position of Current-  
Modifiee;  
SREP[POS, 2] =  
Address of found  
relation;  
Go to BACK:] --> 60853
    60872[60872  
Adj-Check = True;  
Cur-Typ = @HEAD;  
Current-Head = First noun in  
invocation modification set;  
Current-Word = ADJ;  
Go to 60111;] --> 60874
    60874{60874  
If  
Adj-Find  
= True} -- F --> 60878
    60874 -- T --> 60876
    60876[60876  
Adj-Prep-Status =  
FOUND-IN-90;  
Adj-Check = False;] --> 60878
    60878{60878  
If there  
is an untried  
noun  
in the invocation  
modification  
set} -- T --> 60880
    60878 -- F --> 60884
    60880[60880  
Adj-Prep-Status =  
NOT-IN-90;  
Adj-Check = False;] --> 60884
    60884[60884  
ReturnADJ-PREP[  
Cur-Nat-Lang, SUBJ,  
ADJ, Cur-Prep,  
A-Sense[Cur-Sense],  
Return-60,  
Adj-Prep-Status,  
Current-Relation];] --> 60853
    60882[60882  
Current-Head = Next  
untried noun in  
invocation  
modification  
set;  
Go to 60111;] --> 60853

```

FIG. 17BB

FIG. 17BB

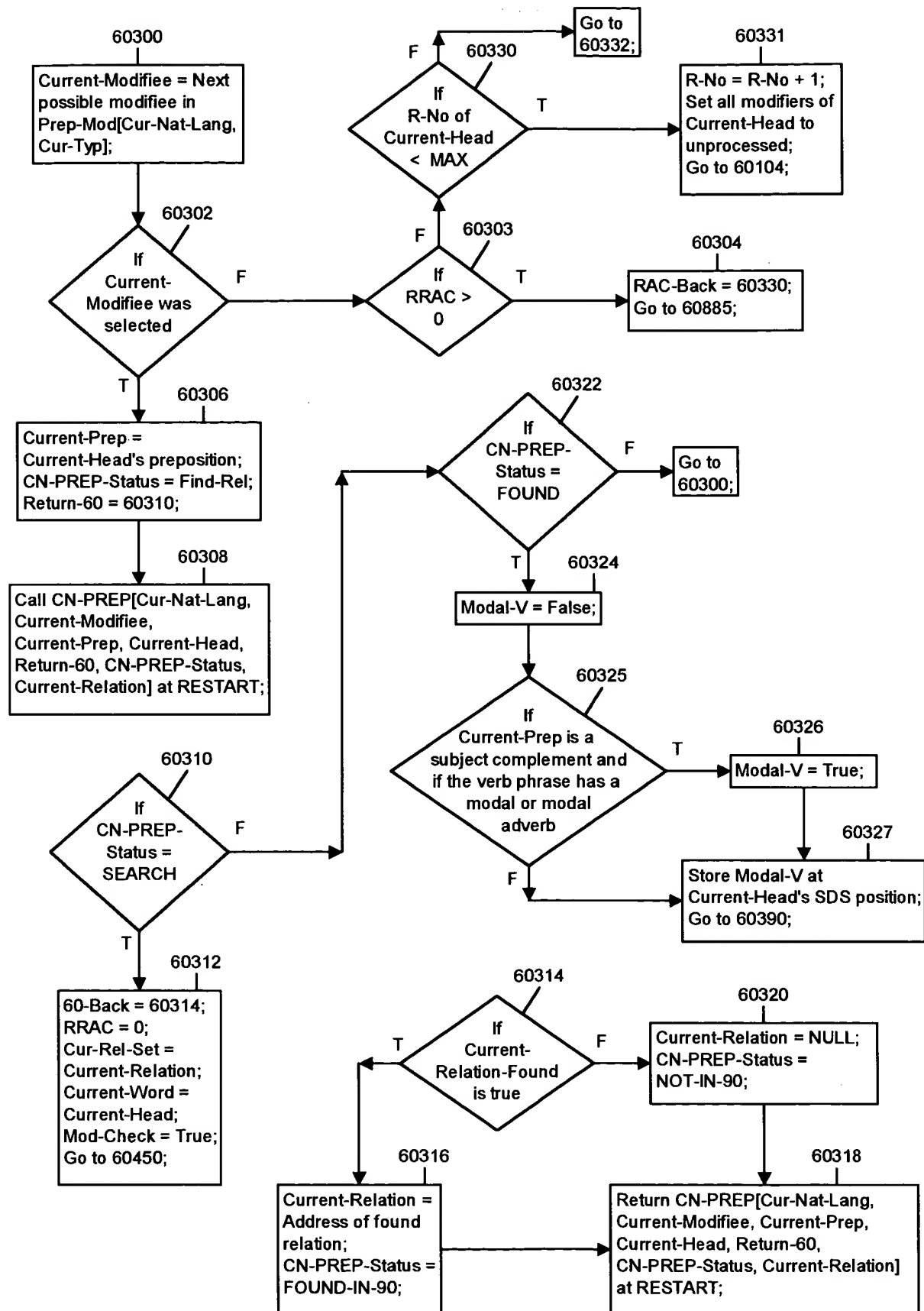


FIG. 17CC

```

graph TD
    60332{60332  
If  
Current-Prep  
can modify the  
clause verb}
    60333[60333  
ADV-Status = 60-Find;  
R-No = L = I = 1;]
    60336{60336  
If  
R-List[R-No]  
meets the source  
requirements of  
ADV-Subclass[  
Current-Prep,  
L]}
    60338[60338  
SUBCLASS[I] =  
ADV-Subclass[Current-Prep, L];  
I = I + 1;  
L = L + 1;]
    60340{60340  
If  
ADV-  
Subclass has an  
untried adverbial  
subclass for  
Current-  
Prep}
    60342{60342  
If  
I > 1}
    60344{60344  
If  
R-No < MAX}
    60346[60346  
R-No = R-No + 1;  
L = 1;]
    60343[60343  
RES = FOUND;]
    60345[60345  
RES = NOT-FOUND;]
    60347{60347  
If  
R-No = 1}
    60348{60348  
If  
ADV-Status  
= 60-Find}
    60351{60351  
If  
RES =  
FOUND}
    60352[60352  
Store at Current-Prep's  
SDS location:  
PREPROC-VERB,  
SUBCLASS, I-1, R-No;  
Go to 60618;]
    60350[60350  
I - I - 1;  
60-Return = 60354;  
Call 70[70-Find,  
Current-Prep, Cur-Rel,  
R-List[R-No], R-No,  
SUBCLASS, I, 60-Return,  
RES, ADV-Status];]
    60354{60354  
If  
RES =  
FOUND}
    60356[60356  
Current-Head = Head of  
Current-Prep's complement;  
Set all modifiers of Current-  
Head to unprocessed;  
Cur-Typ = @VERB-70;  
REQ = Requirements of  
invocation subclass parameter;  
BACK = 60359;  
Go to 60390;]
    60359[60359  
BACK = 60104;  
Go to 60294;]
    60358[60358  
Restore R-No, ADV-Status,  
Current-Prep;  
I = 1;  
P-ADV = False;]
    60349[60349  
Current-Head = Head of  
Current-Prep's  
complement;  
L = 1;  
P-ADV = True;  
Go to 60100;]

    60332 -- T --> 60333
    60332 -- F --> 60353[60353  
Current-Word =  
Current-Head;  
Go to 60355;]
    60333 --> 60336
    60336 -- T --> 60338
    60336 -- F --> 60339[60339  
L = L + 1;]
    60338 --> 60340
    60340 -- T --> 60336
    60340 -- F --> 60342
    60342 -- T --> 60343
    60342 -- F --> 60344
    60344 -- T --> 60346
    60344 -- F --> 60345
    60346 --> 60336
    60343 --> 60351
    60345 --> 60351
    60351 -- T --> 60352
    60351 -- F --> 60348
    60348 -- T --> 60348
    60348 -- F --> 60350
    60350 --> 60354
    60354 -- T --> 60356
    60354 -- F --> 60354
    60356 --> 60359
    60359 --> 60358
    60358 --> 60347
    60347 -- T --> 60349
    60347 -- F --> 60394[60394]
    60349 --> 60348
    60394 --> 60336

```

FIG. 17DD

```

graph TD
    60361{60361  
If Adj-Check  
is true} -- T --> 60363[60363  
Adj-Find = False;  
Go to 60874;]
    60361 -- F --> 60355{60355  
If Current-Word's  
SDS position has  
ABS-Check}
    60355 -- T --> 607000[607000  
Store Clause-Modifier  
at the SDS position of  
the Current-Word;]
    60355 -- F --> 60360{60360  
If Current-Word  
is ellipted and has  
alternative  
processing}
    607000 --> 60369{60369  
If Current-Word  
is a prepositional  
complement}
    60369 -- T --> 60603[Go to 60603;]
    60369 -- F --> 60104[Go to 60104;]
    60360 -- T --> 60362[60362  
Return-60 = 60364;  
Call ELLIP[RESTART,  
Return-60];]
    60360 -- F --> 60364{60364  
If RES-STATUS  
≠ FAILURE}
    60364 -- T --> 60366[60366  
Set all replaced, processed words  
to PROCESSED in the SDS;  
Set all replaced unprocessed  
words and stated words to  
UNPROCESSED in the SDS;]
    60364 -- F --> 60372{60372  
If Current-Word  
is placed through  
morphological  
processing}
    60366 --> 60368{60368  
If a  
noun in  
N-List is  
replaced}
    60368 -- T --> 60370[60370  
Replace all old noun  
heads in N-List with their  
elliptical replacements;  
Set the 1st to the next  
noun to be processed;  
Go to 6000;]
    60368 -- F --> 60104
    60372 -- T --> 60374{60374  
If there is  
another untried  
morphological  
function}
    60372 -- F --> 60384{60384  
If Current-Word  
is a pronoun}
    60374 -- T --> 60376[60376  
RESTART = Address at morphological  
word's SDS position;  
P-Type = INVOCATION-RETURN;  
BASE = Base word of morphological word;  
Return-60 = 60377;  
Call MORPH[RESTART, P-Type, BASE,  
Return-60];]
    60374 -- F --> 60365{60365  
If Current-Word  
is an AMBIGUOUS  
coordinated  
modifier}
    60376 --> 60377[60377  
Set all replaced word to  
UNPROCESSED;]
    60377 --> 60378{60378  
If a noun in  
N-List is  
replaced}
    60365 -- T --> 60367[60367  
Current-Word = UNAMBIGUOUS;  
All modifiers of the Current-Word =  
UNPROCESSED;  
All elliptical and/or morphological  
alternatives of Current-Word =  
untried;  
Assign Current-Word to modify the  
alternative modifiee;  
Go to 60369;]
    60365 -- F --> 60500[Go to 60500;]
    60378 -- T --> 60380[60380  
Replace all old noun heads  
with morphological  
replacements;  
Set the first replacement to  
the next to be processed;  
Go to 6000;]
    60378 -- F --> 60104
    60384 -- T --> 60960[Go to 60960;]
    60384 -- F --> 60500
  
```

FIG. 17EE

FIG. 17EE

```

graph TD
    60500{If Current-Word is a prepositional phrase}
    60502{If Cur-Typ = @VERB-70}
    60504[Next-M = Prep-Mod-Retry[Cur-Nat-Lang, Cur-Typ];]
    60506{If a Next-M is selected}
    60508[Return to caller;]
    60510{If a possible modifier is suspended}
    60512[Store U-List at complement head's SDS position; Go to 60618;]
    60514{If Current-Word has an unprocessed row in CONF-M}
    60516[1st row of CONF-M = 1st unprocessed row; Next-M = CONF-M[1, 1]; Remove source of 1st row;]
    60518{If R-No of Next-M < MAX}
    60520{If Current-Word is an adjective}
    60522[Next-M = Ad-Mod-Retry[Cur-Nat-Lang, Cur-Typ];]
    60524{If a Next-M is selected}
    60526[Next-M = N-Mod-Retry[Cur-Nat-Lang, Cur-Typ];]
    60527[Current-Head = Next-M; Go to 60232;]
    60528[Set Next-M, all of its direct and indirect modifiers, and Current-Word to UNPROCESSED; BACK = 60104; R-No = R-No + 1;]
    60530{If Next-M is a noun phrase head}
    60532[Current-Word = Next-M; Go to 60104;]
    60534[Current-Head = Next-M;]
    60536{If Next-M is a prepositional complement}
    60539{If Next-M is in a clause with a subject complement}
    60540[R-No = R-No + 1;]
    60541{If Next-M is a subject}
    60544{If subject and subject complement have the same head}
    60546[Set no match at 60294; Go to 60295;]

    60500 -- T --> 60502
    60500 -- F --> 60514
    60502 -- T --> 60508
    60502 -- F --> 60504
    60504 --> 60506
    60506 -- T --> 60506
    60506 -- F --> 60510
    60510 -- T --> 60512
    60510 -- F --> 60550
    60514 -- T --> 60516
    60514 -- F --> 60520
    60516 --> 60518
    60518 -- T --> 60524
    60518 -- F --> 60514
    60520 -- T --> 60522
    60520 -- F --> 60526
    60522 --> 60524
    60524 -- T --> 60527
    60524 -- F --> 60550
    60527 --> 60527
    60527 --> 60525
    60525 -- T --> 60527
    60525 -- F --> 60518
    60528 --> 60530
    60530 -- T --> 60534
    60530 -- F --> 60532
    60534 --> 60536
    60536 -- T --> 60536
    60536 -- F --> 60539
    60539 -- T --> 60544
    60539 -- F --> 60540
    60540 --> 60541
    60541 -- T --> 60214
    60541 -- F --> 60216
    60544 -- T --> 60546
    60544 -- F --> 60541
    60546 --> 60293

```

FIG. 17FF

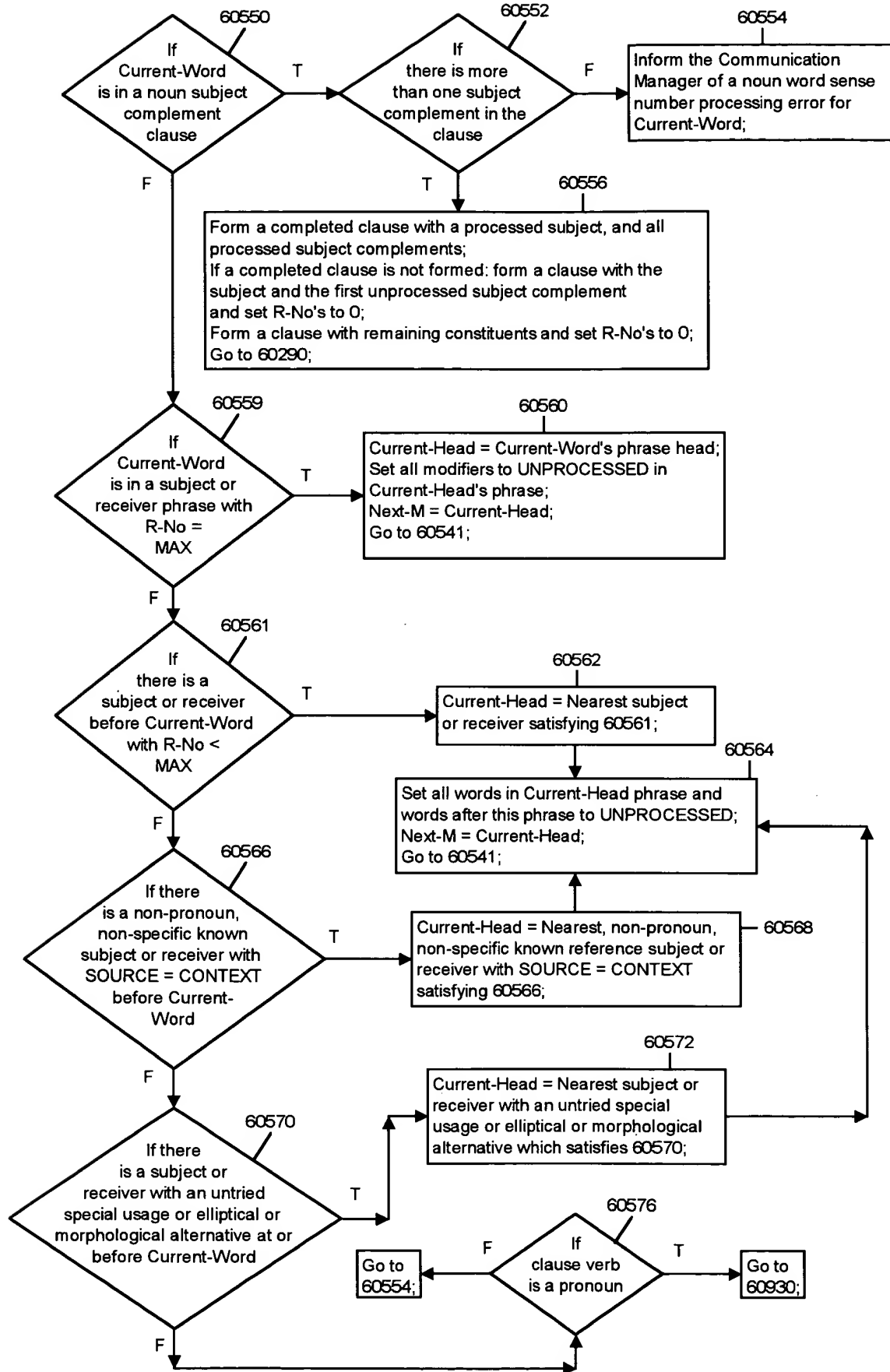


FIG. 17GG

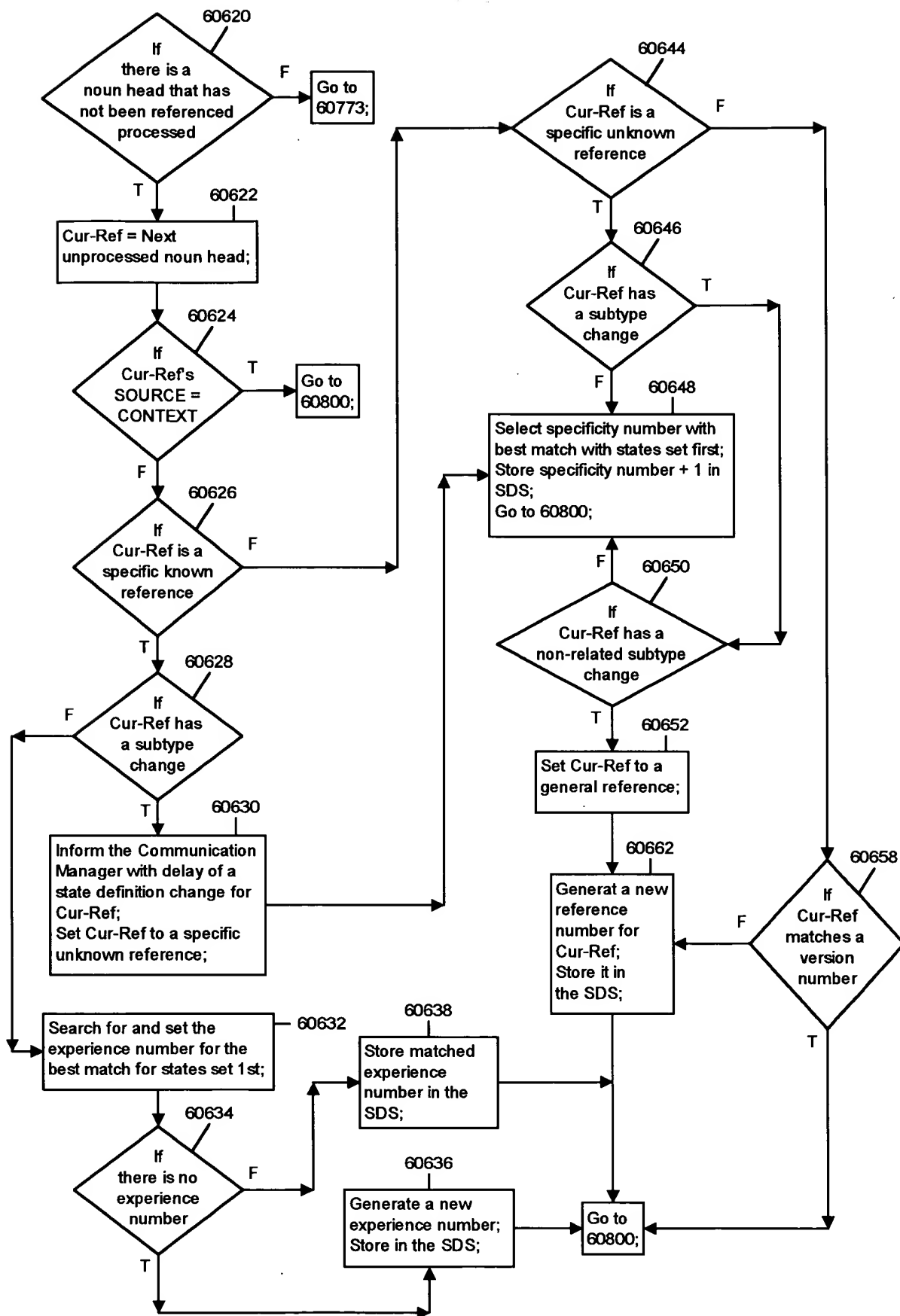


FIG. 17HH

```

graph TD
    60800[60800: Check for Cur-Ref's default reference type for exceptions in Ref-Exception[Cur-Nat-Lang]; Adjust Cur-Ref's reference type as needed;] --> 60802{60802: If Cur-Ref's group descriptor has an ambiguous quantization/ comparison function}
    60802 -- F --> 60812{60812: If Cur-Rel has a special, selection, negation, inclusion, and/or exclusion function}
    60802 -- T --> 60804{60804: If state of compared adjective is stated again in the sentence or in 120}
    60804 -- F --> 60806[60806: Replace ambiguous function with the comparison function and an UNAMBIGUOUS-BY-ASSUMPTION mark;]
    60804 -- T --> 60808{60808: If Cur-Ref can be quantized}
    60808 -- F --> 60812
    60808 -- T --> 60810[60810: Replace ambiguous function with the quantization function and an UNAMBIGUOUS-BY-ASSUMPTION mark;]
    60810 --> 60812
    60812 -- F --> 60816{60816: If Cur-Rel has an unambiguous quantization function}
    60812 -- T --> 60814[60814: Evaluate the functions; Store results in the SDS;]
    60816 -- F --> 60820{60820: If Cur-Rel has an unambiguous comparison function}
    60816 -- T --> 60818[60818: Evaluate the function and store the results in the SDS;]
    60820 -- F --> 60836{60836: If Cur-Rel group descriptor differs from 120}
    60820 -- T --> 60822{60822: If the C-Relation is stored at a compared element}
    60822 -- F --> 60832[60832: Generate and store the C-relation in the SDS;]
    60822 -- T --> 60824{60824: If the C-relation is consistent}
    60824 -- F --> 60826{60826: If compared elements are properties}
    60824 -- T --> 60832
    60826 -- F --> 60830[60830: Inform the Communication Manager with delay of a compared state inconsistency for Cur-Rel;]
    60826 -- T --> 60828[60828: Inform the Communication Manager with delay of a compared property inconsistency for Cur-Rel;]
    60830 --> 60836
    60828 --> 60836
    60836 -- T --> 60838[60838: Mark group descriptor NEW; Go to 60620;]
    60836 -- F --> 60840[60840: Mark the group descriptor with a pointer to the match; Go to 60620;]
  
```

FIG. 17II

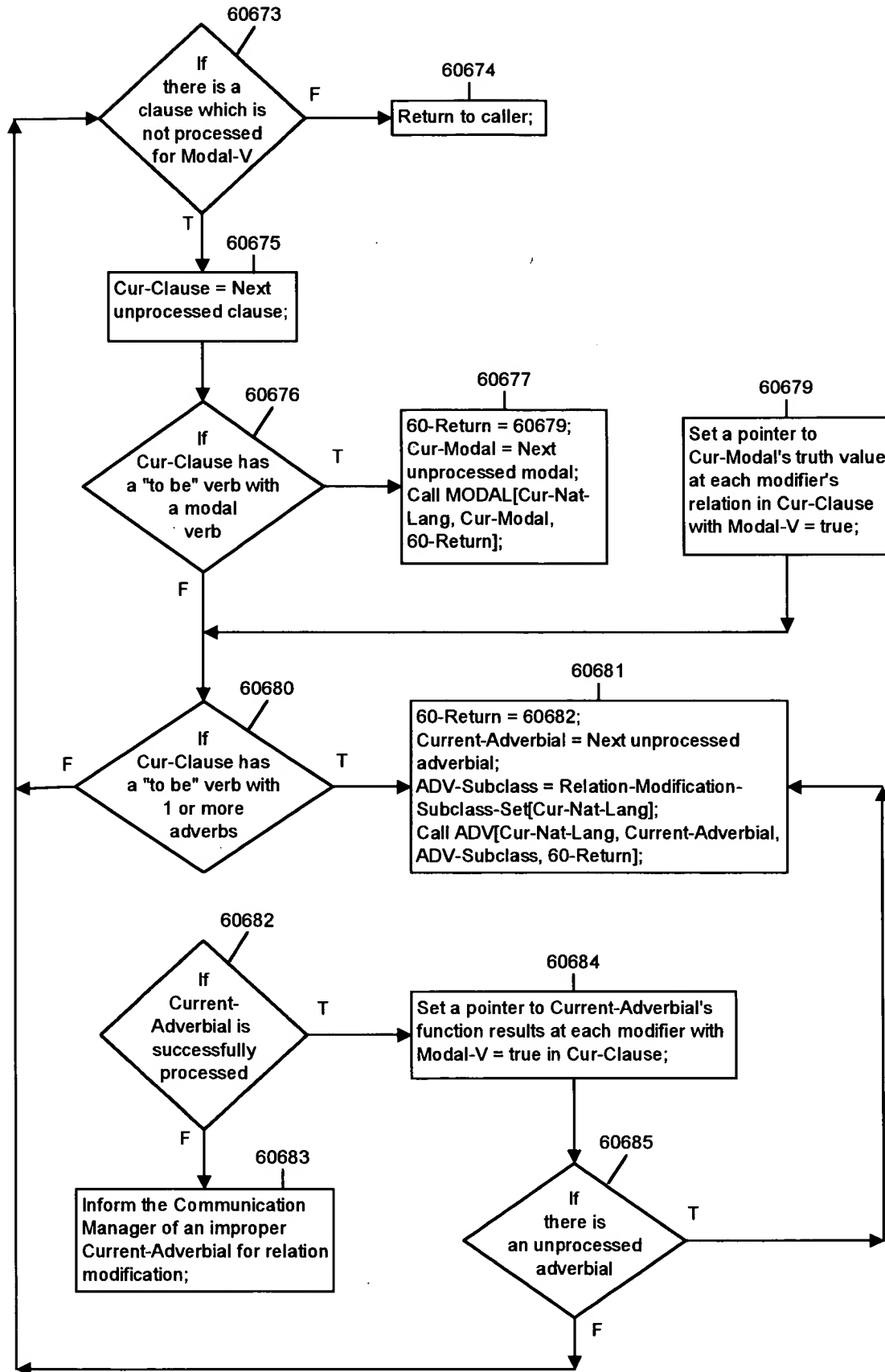


FIG. 17JJ

007260-08572960

DIRECT CATEGORY FORMAT:
<p>List of:</p> <ul style="list-style-type: none">Concrete noun word sense numbers,State abstract noun word sense numbers,Clausal abstract noun word sense numbers,Clausal abstract noun direct category pointer, and/orClausal abstract noun indirect category pointer;

FIG. 18A

INDIRECT CATEGORY FORMAT:**List of descriptors:**

- a) Each descriptor contains one or more adjective or state abstract noun word sense numbers with owners;
- b) Each adjective word sense identifying number contains a state number and a value;
- c) Each value is a typical value, a specific value, or a value range;
- d) A descriptor may have a concrete noun or abstract noun word sense identifying number;

FIG. 18B

```

graph TD
    607002{If  
Current-Head's  
SOURCE = CONTEXT  
and has non-general  
category  
referencing} -- T --> G60618[Go to  
60618]
    607002 -- F --> 607004{If  
Current-Head has  
been processed for  
its representational  
referent}
    607004 -- T --> G607030[Go to  
607030]
    607004 -- F --> 607005[Form an additional R-List for the  
Current-Head with its associated  
direct and indirect category elements;  
R-No = 1;  
MAX = Number of R-List entries;  
RMAX = Number of direct category  
elements in R-List;  
S-REQ = Current-Head's requirements  
from the modifying subordinate clause]
    607005 --> 607006{If there  
is an unprocessed  
direct entry in  
R-List}
    607006 -- T --> 607007[Search for a match of next  
direct entry in 120 which  
meets REQ and S-REQ]
    607007 --> 607008{If a  
match is  
found}
    607008 -- F --> 607010[Remove the  
direct entry;  
MAX = MAX - 1]
    607008 -- T --> 607009[Store the matched  
elements;  
Adjust MAX]
    607009 --> 607010
    607010 --> 607012{If  
Current-Head  
has an unprocessed  
indirect entry}
    607012 -- F --> 607024{If an  
direct or indirect  
entry match is  
found}
    607012 -- T --> 607017[Search for a match of next  
indirect entry in 120 which  
meets REQ]
    607017 --> 607019{If a  
match is  
found}
    607019 -- F --> 607023[Remove the  
indirect entry;  
MAX = MAX - 1]
    607019 -- T --> 607021[Store the matched  
elements;  
Adjust MAX]
    607021 --> 607023
    607023 --> 607024
    607024 -- T --> 607027{If  
Current-Head has  
an unprocessed modifier  
with CLAUSE-MODIFIER}
    607024 -- F --> G607026[Go to  
607026]
    607027 -- T --> 607016[Create an SDS position for an element without one;  
Store ABS-Check at Current-Head's SDS position;  
All unprocessed modifiers with CLAUSE-MODIFIER  
are set to modify the Current-Head;  
BACK = 60104;  
Go to 60104]
    607027 -- F --> G607026
    607026[Store at Current-Head's SDS  
position:  
NO-CONTEXT-REFERENT] --> 607012

```

FIG. 18C

```

graph TD
    607030{607030  
If  
there are  
unprocessed modifiers with  
CLAUSE-MODIFIER and  
there are no unprocessed  
possible modifiees}
    607032[607032  
V-W-S = Word sense number of the verb  
of the modifying subordinate clause;  
Ad-Set = Set of unprocessed modifiers;  
RET = 607034;  
Call 70[ABS-MOD, V-W-S, Ad-Set,  
M-Find, RET];]
    607026[607026  
C-ABS-REF = R-List[R-No];  
Store at Current-Head's SDS  
position: ABSTRACT-NOUN-  
REFERENT, R-No, MAX, R-List,  
C-ABS-REF;]
    607038{607038  
If  
there is an  
unprocessed modifying  
subordinate clause  
sentence role with a  
context referent}
    607040[607040  
Current-Head = Next  
unprocessed context  
referent sentence role;  
Go to 607005;]
    607039{607039  
If there  
is an unprocessed  
modifier with CLAUSE-  
MODIFIER and there is  
an unprocessed possible  
modifiee}
    607041[607041  
Current-Head = Next  
unprocessed possible  
modifiee;  
Go to 607016;]
    607042{607042  
If the  
clausal abstract  
noun has a clause  
requiring purpose  
relation processing}
    607046[607046  
Form a Purpose-Set composed of  
purpose relation descriptors associated  
with the clausal abstract noun;  
Store the Purpose-Set at the clausal  
abstract noun's SDS position;  
Go to 60618;]
    607034{607034  
If  
M-Find is  
true}
    607036[607036  
Set all modifiers of the clausal  
abstract noun to  
UNPROCESSED;  
Current-Head = Clausal  
abstract noun;  
Remove CLAUSE-MODIFIER  
from all modifiers;  
Remove created SDS  
positions;]
    607018{607018  
If  
R-No of  
Current-Head <  
MAX}
    607020[607020  
R-No = R-No + 1;  
Next-M = Current-Head;  
Go to 60536;]
    60360[60360  
Go to  
60360;]
    60618[60618  
Go to  
60618;]

    607030 -- T --> 607032
    607030 -- F --> 607026
    607032 --> 607034
    607034 -- T --> 607026
    607034 -- F --> 607036
    607026 --> 607038
    607038 -- T --> 607040
    607038 -- F --> 607039
    607039 -- T --> 607041
    607039 -- F --> 607042
    607042 -- F --> 60618
    607042 -- T --> 607046
    607036 --> 607018
    607018 -- T --> 607020
    607018 -- F --> 60360
    607020 --> 60360
  
```

FIG. 18D

09671580-092700

VERB WORD SENSE NUMBER FORMAT:
<p><u>Word Sense Identification Number:</u></p> <p>Number identifying the related word sense numbers with a: Verb Class Number; Verb Class Member Number; Sentence Role Identification Numbers: Sentence Role Class Number; Sentence Role Class Member Number (optional);</p>
<p><u>Type Number:</u></p> <p>Number identifying a categorization of the word sense number;</p>
<p><u>Specificity Number:</u></p> <p>Zero - General process; Even ($\neq 0$) - Specific known process path; Odd - Specific unknown process path nearest to the preceding specificity number;</p>
<p><u>Experience Number:</u></p> <p>Zero - Typical process for the associated type and specificity number; \neq Zero - Specific process for the associated type and specificity number;</p>

FIG. 19A

AFFECTED ELEMENTS	EFFECT	RESULT TYPE	PURPOSE POINTER	COMMON ADVERBIAL POINTER	TYPICAL PROCESS	PROCESS SELECTING ADVERBIALS
Subject, indirect object, direct object, instrument, etc., or context element	One or more states and associated value set for the corresponding affected sentence role	A value of: STATIVE, EVENTIVE, and/or HABITIVE	Pointer to purposes associated with word senses in Memory 130	Pointer to table of adverbial subclasses shared with other verbs in Memory 100	Most common process implied by the verb word word sense number	List of adverbials stored in 100 for the verb word sense which have a subclass value that selects process(es) of the verb word sense; each adverbial has an associated type number(s)

FIG. 19B

APPLICATION PATTERN FOR EACH VERB WORD SENSE NUMBER:

Sentence patterns include:

FIG. 19C

0306250-00000000

0967150-09208

SENTENCE ROLE	REQUIREMENT DESCRIPTOR	A-RELATION SET
Subject, indirect object, direct object, etc.	A word sense descriptor composed of 1 or more word sense ID numbers, each with a range of allowed type numbers, and/or 1 or more terms of states and/or properties, each state and property has an associated value or value range; the descriptor may be enumerated or represented with an address of an enumeration in a table;	Set of A-Relations which the sentence role can belong to

FIG. 19D

ENTRY NUMBER	SEMANTIC ROLE	SOURCE REQUIREMENT	VALUE DESCRIPTORS
Local entry number	Specific role of the adverbial	1 or more states, properties, parameters, of functions which the adverbial must satisfy	Each descriptor contains: a required value range; a process application vector for each corresponding value range; an optional value range translation function; an optional pointer to purposes related to a value range;

FIG. 19E

004260-08572960

ENTRY NUMBER	JOINT/SEPARATE POINTER	MAIN SENTENCE ROLE PAIRS	ADVERBIAL SUBCLASS TRIPLETS	PROCESS POINTER
Type #, zero Specificity # and Experience #'s	Pointer to joint/separate criteria	Main sentence role and associated entity requirement pairs	Adverbial subclass pointer, subclass value or value range, and requirement number triplets	Pointer to process entry in Memory 130
Type #, Specificity #, zero Experience #		Main sentence role and associated typical entity for Specificity # pairs	Adverbial subclass pointer, typical subclass value for the Specificity #, and requirement number triplets	Pointer to process entry in Memory 130
Type #, Specificity #, Experience #		Main sentence role and associated experienced entity pair	Adverbial subclass pointer, experienced subclass value, and requirement number triplets	Pointer to process entry in Memory 130

FIG. 19F

- **Sentence Role** (e.g., subject, indirect object, direct object)

- Constant Type (@JOINT, @SEPARATE, @INDETERMINATE)

- Adverbial and/or sentence role criteria for selecting a joint or separate clause

FIG. 19G

```

graph TD
    7000{7000  
If current  
invocation opcode  
is DV or  
DV-S} -- T --> 70100[Go to  
70100]
    7000 -- F --> 7004{7004  
If  
current  
invocation opcode  
is R}
    7004 -- T --> 70380[Go to  
70380]
    7004 -- F --> 7008{7008  
If  
current  
invocation opcode  
is ABS-MOD or  
ADJ-COMP-  
MOD}
    7008 -- T --> 70400[Go to  
70400]
    7008 -- F --> 7012{7012  
If  
current  
invocation opcode  
is T-Rel}
    7012 -- T --> 70500[Go to  
70500]
    7012 -- F --> 7016{7016  
If  
current  
invocation opcode  
is  
COMPLETION}
    7016 -- T --> 70700[Go to  
70700]
    7016 -- F --> 7020{7020  
If  
current  
invocation opcode  
is Pre-Selected-  
Word-Sense}
    7020 -- T --> 70800[Go to  
70800]
    7020 -- F --> 7024{7024  
If  
current  
invocation opcode  
is  
70-Find}
    7024 -- T --> 70Return[Go to  
70-Return]
    7024 -- F --> 7028[7028  
(The opcode is S-REQ)  
Set S-REQ to the  
designated sentence  
role  
requirements of the  
designated verb word  
sense;  
Return to caller]
    7028 --> 7020
  
```

```

graph TD
    70100{If current invocation opcode is R} -- T --> 70102[Form TR-List for subject; Form V-List for verb;]
    70100 -- F --> 70101{If clause is preprocessed}
    70101 -- T --> 70130[Go to 70130]
    70101 -- F --> 70120{If Current-Clause has a coordinated sentence role(s) with a respective function}
    70120 -- T --> 70122[Form a separate clause in the SDS for each coordinated constituent in the SDS joined by the conjunctions;]
    70120 -- F --> 70126[70126: Mul-V = True; Cur-Conj-Set = Conjunctions in verb phrase; 70-Return = 70114; Call CONJ[Cur-Nat-Lang, Cur-Conj-Set, 70-Return];]
    70122 --> 70124{If Current-Clause has coordinated verbs}
    70124 -- T --> 70126
    70124 -- F --> 70125[70125: Mul-V = False;]
    70125 --> 70114{If Current-Clause has coordinated subjects}
    70114 -- T --> 70115[70115: Mul-S = True; Cur-Conj-Set = Conjunctions in subject phrase; 70-Return = 70117; Call CONJ[Cur-Nat-Lang, Cur-Conj-Set, 70-Return];]
    70114 -- F --> 70116[70116: Mul-S = False;]
    70116 --> 70117{If Current-Clause has coordinated direct objects}
    70117 -- T --> 70118[70118: Mul-D = True; Cur-Conj-Set = Conjunctions in direct object phrase; 70-Return = 70127; Call CONJ[Cur-Nat-Lang, Cur-Conj-Set, 70-Return];]
    70117 -- F --> 70119[70119: Mul-D = False;]
    70119 --> 70127{If Current-Clause has coordinated indirect objects}
    70127 -- T --> 70128[70128: Mul-I = True; Cur-Conj-Set = Conjunctions in indirect object phrase; 70-Return = 70132; Call CONJ[Cur-Nat-Lang, Cur-Conj-Set, 70-Return];]
    70127 -- F --> 70130
    70128 --> 70129[70129: Mul-I = False;]
    70129 --> 70132[70132: Form an R-List for each subject, indirect and direct object without one; Set R-No's to 1; Form a V-List for each main verb; Store MAX's and UNPREPROCESSED at each of the sentence roles; Form an all 1 VM-V for each verb; F-Stat = SUBJECT; REPROC = False;]
    70132 --> 70133{If Current-Clause has a detectable special usage}
    70133 -- T --> 70136[70136: Perform the functions associated with the special usages in SSU[Cur-Nat-Lang];]
    70133 -- F --> 70200[Go to 70200]
    70136 --> 70138{If Cur-Sub has failed a non-special usage}
    70138 -- T --> 70212[Go to 70212]
    70138 -- F --> 70140[70140: Mark a failed special usage VNU-V as DISALLOWED; Go to 70218;]
    70102 --> 70104[70104: For each verb word sense in V-List until an ID# or requirement match is found: For a word sense in TR-List: Determine if noun word sense ID# matches a subject requirement in V-List;]
    70104 --> 70106{If a match is found}
    70106 -- T --> 70108[70108: Acceptable-Subject = True;]
    70106 -- F --> 70110[70110: Acceptable-Subject = False;]
    70108 --> 70112[70112: Return to caller;]
    70110 --> 70117
  
```

Go to
70200

```

graph TD
    70200[Cur-Sub = Next UNPREPROCESSED subject;] --> 70202{If the Cur-Sub R-No's word sense # matches a possible requirement at each coordinated verb of Cur-Sub}
    70202 -- T --> 70204[Store a zero at each position of a Cur-Sub verb's VM-V with a usage that doesn't match a word sense number associated with Cur-Sub for that verb; Cur-Sub = PREPROCESSED; Store a 1 at each verb's SZ-M which was zeroed; Store SZ-V and normal R-No;]
    70202 -- F --> 70206[Store R-No, # of matches, VNU-V at Cur-Sub's SDS position;]
    70206 --> 70208{If Cur-Sub's R-No < MAX}
    70208 -- T --> 70210[R-No = R-No + 1;]
    70208 -- F --> 70212[Determine if the allowed special usages of Cur-Sub in SSU[Cur-Nat-Lang] imply word senses that match word sense number requirements of unmatched verbs in a VNU-V of an R-No stored for Cur-Sub in the SDS starting with the 1st R-No and its VNU-V;]
    70212 --> 70214{If each unmatched verb has a special usage subject match}
    70214 -- T --> 70216[Evaluate the functions for each special use; Store at Cur-Sub's SDS position: each special use number, each implied subject R-No, each special use verb position, VNU-V;]
    70214 -- F --> 70218{If Cur-Sub's R-No < MAX}
    70218 -- F --> 70262[Go to 70262;]
    70218 -- T --> 70210
    70216 --> 70224{If a subject is UNPREPROCESSED}
    70224 -- F --> 70150[Go to 70150;]
    70224 -- T --> 70226{If next UNPREPROCESSED subject has a word sense no. which matches a requirement of a PREPROCESSED subject}
    70226 -- F --> 70200
    70226 -- T --> 70228[Set next UNPREPROCESSED subject to PREPROCESSED; Store at the subject's SDS position: 1st matching R-No, the matched subject's position, VNU-V, special use number, R-No, verb position; Evaluate any special uses;]
    70228 --> 70224

```

FIG. 19J

```

graph TD
    70150{70150  
If clause has an  
UNPREPROCESSED  
indirect object}
    70152[70152  
Cur-I-Obj = Next  
UNPREPROCESSED  
indirect object;  
Cur-I-Obj's R-No = 1;]
    70154{70154  
If  
Cur-I-Obj's  
R-List[R-No] word  
sense no. matches  
1 or more possible  
requirements for  
each verb of  
Cur-I-Obj}
    70162[70162  
Store Cur-I-Obj's normal R-No;  
Store a zero at each position of  
a Cur-I-Obj verb's VM-V with a  
usage that doesn't match a  
word sense number of  
Cur-I-Obj for that verb;  
Store a 1 at each verb's IOZ-V  
which was zeroed;  
Store IOZ-V at Cur-I-Obj;  
Cur-I-Obj = PREPROCESSED;]
    70164[70164  
Store R-No, number of  
matches, VNU-V at  
Cur-I-Obj's SDS position;  
Go to 70156;]
    70158[70158  
Increment  
Cur-I-Obj's  
R-No by 1;]
    70156{70156  
If  
Cur-I-Obj's  
R-No <  
MAX}
    70160[70160  
F-Stat = INDIRECT-  
OBJECT;  
Go to 70250;]
    70308{70308  
If  
Cur-D-Obj's  
R-No <  
MAX}
    70310[70310  
Increment  
Cur-D-Obj's  
R-No by 1;]
    70312[70312  
F-Stat = DIRECT-  
OBJECT;  
Go to 70290;]
    70314[70314  
Store R-No,  
number of  
matches, VNU-V at  
Cur-D-Obj's SDS  
position;  
Go to 70308;]
    70304{70304  
If  
Cur-D-Obj's  
R-List[R-No] word  
sense no. matches  
1 or more possible  
requirements for  
each verb of  
Cur-D-Obj}
    70306[70306  
Store Cur-D-Obj's normal R-No;  
Store a zero at each position of a  
Cur-D-Obj verb's VM-V with a  
usage that doesn't match a word  
sense number of Cur-D-Obj for  
that verb;  
Store a 1 at each verb's IOZ-V  
which was zeroed;  
Store DOZ-V at Cur-D-Obj;  
Cur-D-Obj = PREPROCESSED;  
Go to 60316;]
    70300{70300  
If  
clause has an  
UNPREPROCESSED  
direct object}
    70302[70302  
Cur-D-Obj = Next  
UNPREPROCESSED  
indirect object;  
Cur-D-Obj's R-No = 1;]
    70360[70360  
Go to 70360;]
    70166{70166  
If  
there is an  
UNPREPROCESSED  
indirect object}
    70168{70168  
If  
next  
UNPREPROCESSED  
indirect object has a word  
sense ID# which matches a  
word sense no. requirement  
of a PREPROCESSED  
indirect object}
    70170[70170  
Set the next UNPREPROCESSED  
indirect object to PREPROCESSED;  
Store at this indirect object's SDS  
position: the 1st matching R-No,  
the matched indirect object's  
position, VNU-V, special use  
numbers, R-No's and verb  
positions;  
Evaluate special use functions;]

    70150 -- T --> 70152
    70150 -- F --> 70158
    70152 --> 70154
    70154 -- T --> 70162
    70154 -- F --> 70164
    70162 --> 70166
    70164 --> 70156
    70158 --> 70156
    70156 -- T --> 70158
    70156 -- F --> 70160
    70160 --> 70308
    70308 -- T --> 70310
    70308 -- F --> 70312
    70310 --> 70304
    70304 -- T --> 70306
    70304 -- F --> 70314
    70306 --> 70300
    70300 -- T --> 70302
    70300 -- F --> 70360
    70302 --> 70304
    70166 -- T --> 70168
    70166 -- F --> 70158
    70168 -- T --> 70170
    70168 -- F --> 70152
    70170 --> 70152
  
```

FIG. 19K

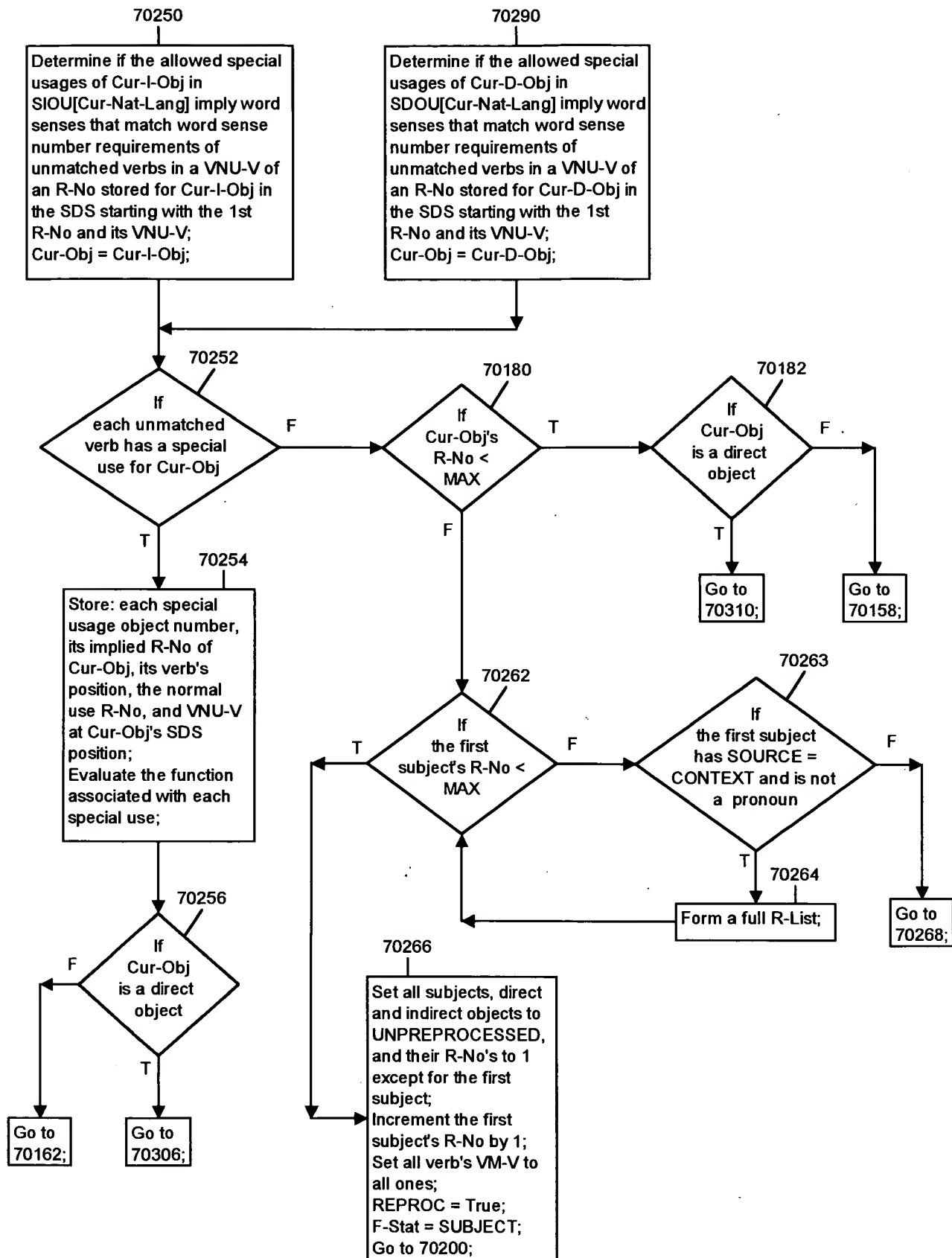


FIG. 19L

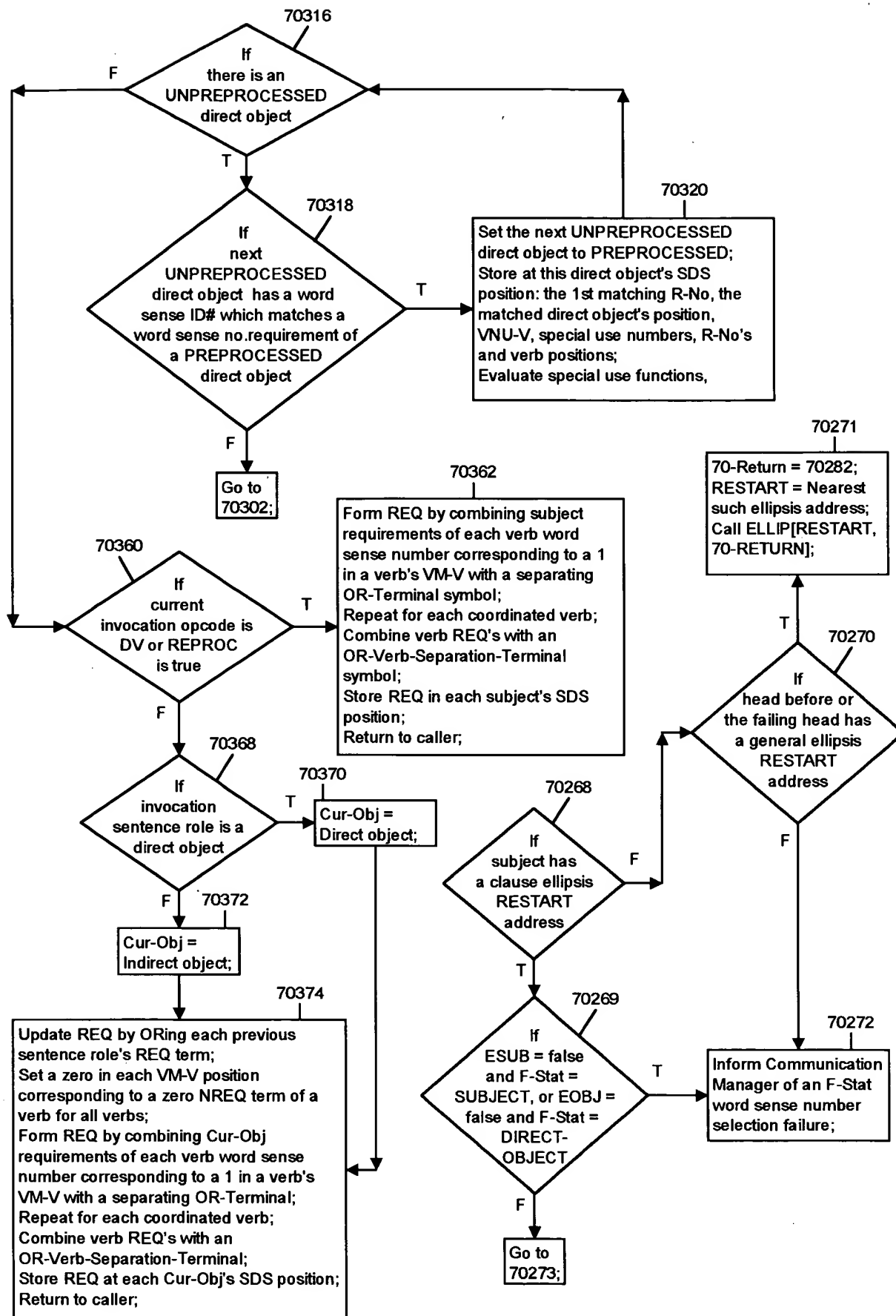


FIG. 19M

FIG. 19N

```

graph TD
    70400[70400  
Cur-Mod = Next UNPREPROCESSED  
modifier in the invocation modification set;  
BASE = Base word of Cur-Mod;  
AFFIX = Affixes of Cur-Mod or NULL;  
SOURCE = Part of speech of BASE;  
DESTINATION = ADVERBIAL;  
70-Return = 70402;  
P-Type = GENERATE;  
Call MORPH[Cur-Nat-Lang, P-Type,  
BASE, AFFIX, SOURCE, DESTINATION,  
70-Return];] --> 70408{70408  
If  
invocation  
modification set has an  
UNPREPROCESSED  
modifier}
    70408 -- T --> 70400
    70408 -- F --> 70409[70409  
Cur-V-W-S = Word  
sense number of  
invocation verb;]
    70409 --> 70410{70410  
If  
invocation  
modification set has an  
UNPROCESSED  
modifier}
    70410 -- T --> 70414[70414  
Cur-Mod = Next  
UNPROCESSED  
modifier in the  
invocation  
modification set;]
    70410 -- F --> 70417[70417  
Verb-W-S = Cur-V-W-S;  
70-Back = 70412;  
Go to 70620;]
    70417 --> 70408
    70414 --> 70416{70416  
If  
Cur-Mod has an  
unevaluated function  
type}
    70416 -- T --> 70418[70418  
RESTART = Address at SDS  
position of Cur-Mod;  
P-Type = INVOCATION-  
RETURN;  
BASE = Base word of Cur-Mod;  
70-Return = 70420;  
Call MORPH[RESTART, P-Type,  
BASE, 70-Return];]
    70416 -- F --> 70424{70424  
If a  
match is  
found}
    70418 --> 70420{70420  
If  
RESULT-TYPE  
= ADDRESS-  
DESCRIPTOR}
    70420 -- T --> 70422[70422  
Determine if RESULT matches  
an adverbial subclass of  
Cur-V-W-S;]
    70420 -- F --> 70430{70430  
If  
RESULT-TYPE  
= PHRASE}
    70422 --> 70424
    70430 -- T --> 70432[70432  
Go to  
70432;]
    70430 -- F --> 70450[70450  
Cur-Clause = RESULT;  
P-Type = PROCESS-CLAUSE;  
70-Return = 70452;  
Call 18[Cur-Nat-Lang, P-Type,  
70-Return];]
    70450 --> 70452{70452  
If  
RESULT  
is successfully  
processed}
    70452 -- T --> 70454[70454  
Set RESULT to modify the  
invocation verb through the  
implied conjunction;  
Go to 70410;]
    70452 -- F --> 70416
    70424 -- T --> 70426[70426  
Set Cur-Mod to modify the  
invocation verb with the  
matched RESULT subclass;  
Go to 70410;]
    70424 -- F --> 70416
    70402{70402  
If  
SDS position  
of Cur-Mod has  
FAIL} -- T --> 70404[70404  
M-Find = False;  
Return to caller;]
    70402 -- F --> 70406[70406  
Cur-Mod is set to  
PREPROCESSED;]
    70406 --> 70408

```

FIG. 190

```

graph TD
    70432{If RESULT is a prepositional phrase}
    70433[The RESULT prepositional complement's R-No = 1; R-Lis[R-No] = NULL;]
    70434[70-Return = 70436; Current-Prep = Preposition of RESULT; Cur-Rel = NULL; SUBCLASS = NULL; I = 1; 60-Start = 60354; RES = PREP-COMP; ADV-Status = 70-FIND; Call 60[Current-Prep, Cur-Rel, R-Lis[R-No], R-No, SUBCLASS, I, 60-Start, RES, ADV-Status];]
    70442[Current-Adverbial = RESULT; Verb-Subclass = Adverb subclasses of Cur-V-W-S; 70Return= 70444; Call ADV[Cur-Nat-Lang, Current-Adverbial, Verb-Subclass, 70-Return];]
    70444{If RESULT is successfully processed}
    70436{If RES = FOUND}
    70440[Store SUBCLASS at prepositional complement of RESULT; Current-Adverbial = RESULT; Verb-Subclass = Prepositional subclasses of Cur-V-W-S; 70-Return = 70460; Call ADV[Cur-Nat-Lang, Current-Adverbial, Verb-Subclass, 70-Return];]
    70460{If RESULT is successfully processed}
    70462{If R-No of prepositional complement < MAX}
    70470{If there are other adverbial interpretations}
    70464[SUBCLASS = Adverbial subclass requirements of subclass selected at ADV; 70-Return 70466; 60-Start = 60354; RES = FOUND; ADV-Status = 70-FIND; Call 60[Current-Prep, SUBCLASS, 60-Start, RES, 70-Return, ADV-Status];]
    70466{If RESULT's complement is fully processed for its word sense number}
    70472[70-Return = 70460; Call ADV[RESTART, Current-Adverbial, Verb-Subclass, 70-Return];]

    70432 -- T --> 70433
    70432 -- F --> 70442
    70433 --> 70434
    70434 --> 70436
    70436 -- T --> 70440
    70436 -- F --> 70416[Go to 70416]
    70440 --> 70444
    70444 -- T --> 70426[Go to 70426]
    70444 -- F --> 70416
    70416 --> 70460
    70460 -- T --> 70464
    70460 -- F --> 70462
    70462 -- T --> 70434
    70462 -- F --> 70470
    70470 -- T --> 70472
    70470 -- F --> 70416
    70472 --> 70466
    70466 -- T --> 70426
    70466 -- F --> 70462
    70464 --> 70466
  
```

FIG. 19P

```

graph TD
    70500[T-Desc = NULL;  
Re-ADV-S = False;  
Separate multiple  
T-Relations into separate  
sentences;  
T-REL = T-Relation of the  
1st sentence;] --> 70502{If  
adjective has a  
state representation}
    70502 -- F --> 70560[Go to  
70560;]
    70502 -- T --> 70504[Combine subclasses of  
source and destination  
clause verbs' word sense  
numbers that have the  
same semantic roles into  
Com-Sub;]
    70504 --> 70506{If  
Com-Sub  
is not  
empty}
    70506 -- T --> 70518[Invocation-Modification-Set = Adjective;  
Invocation-Verb = Source clause verb;  
Invocation-Verb-W-S-Subclass = Pointer to  
Com-Sub;  
Invocation-Opcode = ADJ-COMP-MOD;  
Caller-Return = 70520;  
Call 70[Invocation-Opcode, Invocation-  
Modification-Set, Invocation-Verb,  
Invocation-Verb-W-S-Subclass, M-Find,  
Caller-Return];]
    70506 -- F --> 70508{If  
destination  
clause verb has an  
untried word sense  
number  
combination}
    70508 -- T --> 70510[Set destination clause verb word  
sense numbers to the next  
untried word sense number  
combination;  
Set destination adverbials to  
UNPROCESSED;]
    70508 -- F --> 70512{If  
source  
clause verb has an  
untried word sense  
number  
combination}
    70512 -- T --> 70514[Set source clause verb word sense numbers to  
the next untried word sense number combination;  
Set source adverbials to UNPROCESSED;  
Re-ADV-S = True;  
Set destination clause verb word sense numbers  
to the first word sense number;  
Set all other destination clause word sense  
numbers to untried;  
Set destination adverbials to PROCESSED;]
    70512 -- F --> 70516[Append to T-Desc: NEGATIVE-  
ADJ-TO-ADV-T-RELATION;  
Restore initial verb word sense  
numbers;  
Go to 70560;]
    70518 --> 70520{If  
M-Find  
is true}
    70520 -- T --> 70526[Go to  
70526;]
    70520 -- F --> 70504
    70510 --> 70504
    70514 --> 70504

```

FIG. 19Q

FIG. 19R

```

graph TD
    70700[70700  
CS = CSD = MVS = 0; V = S = 1;  
VAG = Number of verbs;  
Cur-Verb = First verb;  
Cur-S-AND = First AND-group of subjects;  
SAG = Number of subject AND-groups;  
Cur-IO-AND = First AND-group of indirect objects;  
IOAG = Number of indirect object AND-groups;  
IO = MIN[IOAG, 1];  
Cur-DO-AND = First AND-group of direct objects;  
DOAG = Number of direct object AND-groups;  
DO = MIN[DOAG, 1]; AMB = False;  
Ellip-Clause-Comb = False; Pre-ADV = False;] --> 70702
    70702[70702  
AND REQ terms of Cur-Verb for  
each sentence role in Cur-S-AND,  
Cur-IO-AND, and Cur-DO-AND to  
form an AND-Result-Vector;] --> 70704
    70704{70704  
If the  
AND-Result-  
Vector is all  
zeroes} -- T --> 70706
    70704 -- F --> 70734
    70706{70706  
If a  
sentence role  
AND-group has  
an ambiguous  
constituent} -- T --> 70708
    70706 -- F --> 70710
    70708[70708  
Separate and remove  
each ambiguous  
constituent;  
Identify and store each  
removed constituent;  
AMB = True;] --> 70702
    70710{70710  
If  
AMB is  
true} -- T --> 70712
    70710 -- F --> 70736
    70712[70712  
Replace removed  
constituents;  
AMB = False;] --> 70714
    70714{70714  
If a  
hypothetical  
truth value for Cur-Verb  
is implied} -- T --> 70720
    70714 -- F --> 70716
    70716{70716  
If  
Cur-Verb has a  
modal} -- T --> 70718
    70716 -- F --> 70717
    70718[70718  
70-Return = 70720;  
Call MODAL[Cur-Nat-  
Lang, Cur-Verb,  
Cur-Modal, 70-Return];] --> 70717
    70717{70717  
If  
Cur-Verb has an  
adverbial with a modal  
semantic role} -- T --> 70719
    70717 -- F --> 70714
    70719[70719  
Current-Adverbial =  
Such an adverbial;  
70-Back = 70720;  
Verb-Subclass = Modal  
subclasses of Cur-Verb;  
Go to 70844;] --> 70720
    70720{70720  
If  
truth value of  
Cur-Verb is set to  
less than true} -- T --> 70724
    70720 -- F --> 70750
    70724[70724  
Store at SDS:  
@HYPO/NOT-  
POSSIBLE-NOW;] --> 70736
    70734{70734  
If  
AMB is  
true} -- T --> 70736
    70734 -- F --> 70738
    70736[70736  
AND each removed constituent  
REQ with AND-Result-Vector;  
Replace removed constituents  
which do not zero  
AND-Result-Vector;  
Assign other removed  
constituents to other groups;  
Set all removed constituents to  
UNAMBIGUOUS;  
AMB = False;] --> 70728
    70738[70738  
Clause-M[S, V, IO, DO, 2]  
= @TRUE;] --> 70728
    70728{70728  
If  
[S, V, IO, D] =  
[SAG, VAG, IOAG,  
DOAG]} -- T --> 70780
    70728 -- F --> 70730
    70730[70730  
Select next untried  
combination of main  
sentence role  
AND-groups and verb;  
Set S, V, IO and D  
according to the selected  
combination;  
Reassign Cur-S-AND,  
Cur-Verb Cur-IO-AND,  
and Cur-DO-AND as  
needed;] --> 70702
    70780[70780  
Go to  
70800;]

```

FIG. 19S

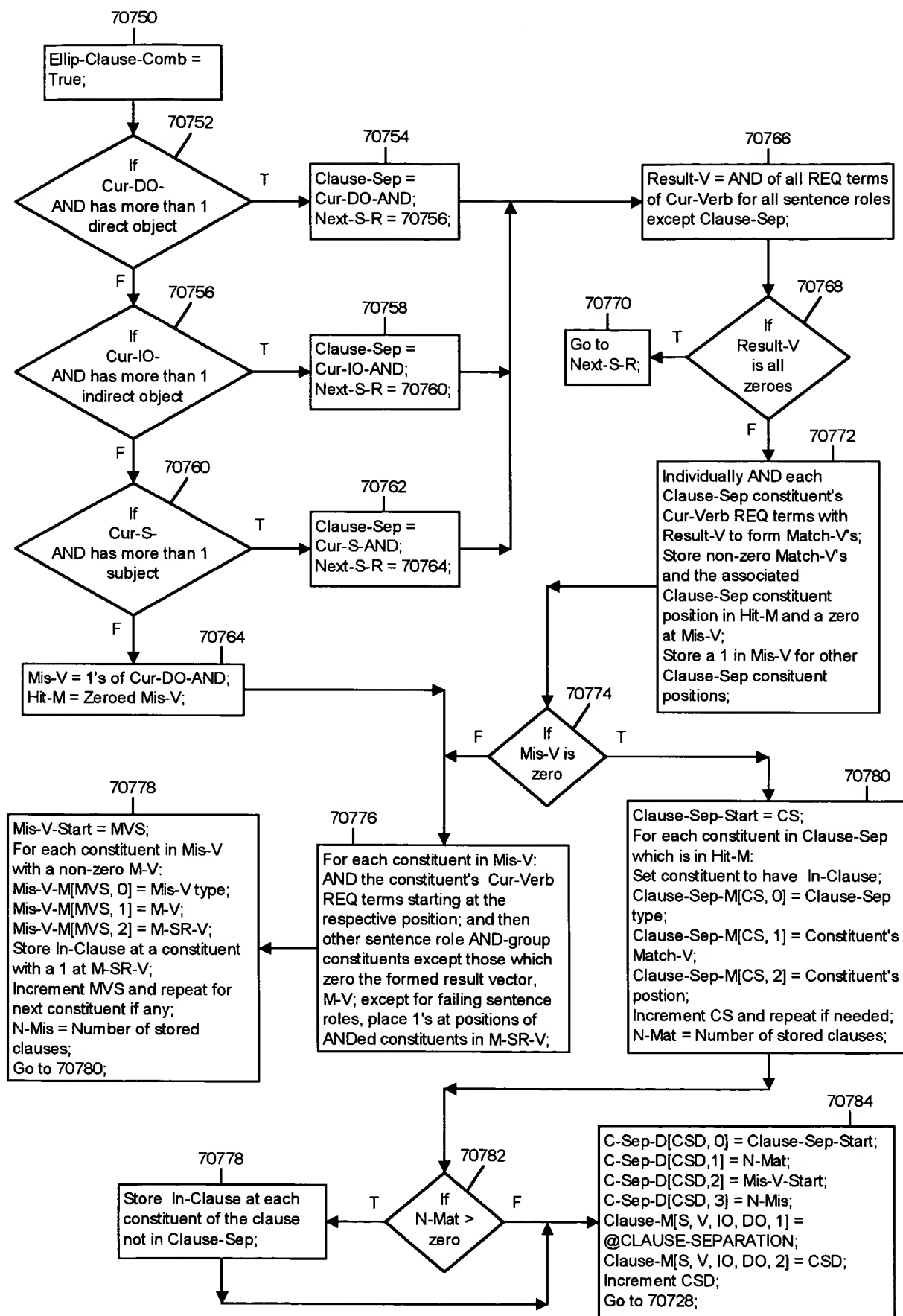


FIG. 19T


```

graph TD
    70800[70800 Cur-Verb = 1st verb of the sentence;  
C-Success = False;] --> 70802{70802 If clause  
containing Cur-Verb  
has pre-selected word-  
sense numbers}
    70802 -- T --> 70839[70839 Cur-Result-V =  
Cur-Verb word sense  
number of next  
UNPROCESSED  
clause;]
    70802 -- F --> 70804{70804 If  
Cur-Verb has  
> 1 Clause-M rows with  
AND-Result-  
Vectors}
    70804 -- T --> 70805{70805 If  
all rows are  
unprocessed}
    70805 -- T --> 70806[70806 For each Clause-M row of  
Cur-Verb with a non-zero  
AND-Result-Vector: AND each  
row's AND-Result-Vector to  
form Cur-Result-V;]
    70805 -- F --> 70808{70808 If  
Cur-Result-V  
is non-zero}
    70808 -- T --> 70810[70810 Set all Clause-M rows  
ANDED at 70806 to  
PROCESSED;  
Store @COMBINED at  
the SDS;]
    70808 -- F --> 70812{70812 If  
Clause-M has  
an UNPROCESSED row  
without @CLAUSE-  
SEPARATION}
    70812 -- T --> 70814[70814 Cur-Result-V = AND-  
Result-Vector of  
such a next  
UNPROCESSED  
Cur-Verb row;  
Set this row to  
PROCESSED;]
    70812 -- F --> 70818[70818 Cur-Result-V = Match-V or M-V of  
next separated clause;  
Set this clause to UNPROCESSED;]
    70810 --> 70822{70822 If  
there is an  
UNPROCESSED  
clause}
    70814 --> 70822
    70818 --> 70822
    70822 -- T --> 70824[70824 Cur-Verb = Next  
UNPROCESSED  
verb;  
Go to 70802]
    70822 -- F --> 70830[70830 Verb-W-S = 1st  
word sense number  
of Cur-Result-V;]
    70830 --> 70831[70831 70-Back = 70840;  
Go to 70600;]
    70831 --> 70850[70850 Verb-Subclass =  
Prepositional  
subclasses of  
Cur-Verb;  
Store  
Verb-Subclass at  
Current-Prep's  
complement;  
Go to 70860;]
    70850 --> 70848{70848 If  
RES =  
FOUND}
    70848 -- T --> 70850
    70848 -- F --> 70857{70857 If  
Pre-ADV  
is true}
    70857 -- T --> 70857F[70857F Pre-ADV = False;  
Go to 70-Back;]
    70857 -- F --> 70858[70858 Inform the  
Communication  
Manager of an  
adverbial  
preposition  
processing error  
for Current-  
Adverbial;]
    70858 --> 70844{70844 If  
Current-  
Adverbial is a  
prepositional  
phrase}
    70844 -- T --> 70845{70845 If  
preposition  
has PREPROC-  
VERB}
    70845 -- T --> 70850
    70845 -- F --> 70846[70846 70-Return = 70848;  
Current-Prep =  
Preposition of  
Current-Adverbial;  
Cur-Rel = NULL;  
R-No = 1;  
R-List[1] = NULL;  
SUBCLASS = NULL;  
I = 1;  
60-Start = 60354;  
RES = PREP-COMP;  
ADV-Status = 70-FIND;  
Call 60[Current-Prep,  
Cur-Rel, R-No,  
R-List[R-No],  
SUBCLASS, I, 60-Start,  
RES, ADV-Status;]
    70846 --> 70851{70851 If  
Current-  
Adverbial is a  
clause}
    70851 -- T --> 70895[70895 Go to]
    70851 -- F --> 70852[70852 Verb-Subclass =  
Adverb subclasses  
of Cur-Verb;  
Store  
Verb-Subclass at  
Current-Adverbial;]
    70852 --> 70860[70860 Go to]
  
```

FIG. 19U

FIG. 19U

```

graph TD
    70600{70600  
If  
UNPROCESSED  
adverbials of  
Verb-W-S are  
coordinated}
    70602[70602  
Cur-Conj-Set = SDS positions of  
conjunctions of such adverbials;  
70-Return = 70604;  
Call CONJ[Cur-Nat-Lang,  
Cur-Conj-Set, 70-Return];]
    70604[70604  
Compute sums of products of  
multi-level conjunctions;  
Form an additional clause of  
the Verb-W-S for each "or"  
conjunction;  
Join each such clause with an  
"or" conjunction;  
Assign an AND-group of  
adverbials to each replicated  
clause;  
Go to 70-Back;]
    70606[70606  
Go to  
70-Back;]
    70610{70610  
If  
Current-  
Adverbial is an  
AMBIGUOUS  
coordinated  
modifier}
    70612[70612  
Current-Adverbial = UNAMBIGUOUS;  
Assign Current-Adverbial to the  
alternate modifiee word sense number;  
Reset elliptical and/or morphological  
alternatives;  
Set all modifiers of the  
Current-Adverbial to UNPROCESSED;  
Go to 70838;]
    70620{70620  
If  
Verb-W-S has  
a conflicting adverbial  
set modifying  
Verb-W-S}
    70622[70622  
Form new clauses with the  
same constituents as the  
clause had before the  
conflicting adverbial processing  
except that no new clause has  
a conflicting adverbial;  
Join the new clauses with  
conjunction joining the  
removed conflicting adverbial;]
    70624[70624  
Go to  
70-Back;]
    70630[70630  
T-Cur-Adverbial =  
Current-Adverbial;]
    70634{70634  
If  
T-Cur-Adverbial  
is modified by an  
UNPROCESSED  
adverbial}
    70636[70636  
Verb-Subclass = Adverbial modifier subclasses  
of T-Cur-Adverbial;  
Current-Adverbial = Next UNPROCESSED  
adverbial modifying T-Cur-Adverbial;  
Go to 70844;]

    70600 -- T --> 70602
    70600 -- F --> 70606
    70602 --> 70604
    70604 --> 70606
    70606 --> 70600
    70610 -- T --> 70612
    70610 -- F --> 70606
    70612 --> 70610
    70620 -- T --> 70622
    70620 -- F --> 70624
    70622 --> 70620
    70622 --> 70624
    70624 --> 70620
    70624 --> 70606
    70630 --> 70634
    70634 -- T --> 70636
    70634 -- F --> 70624
    70636 --> 70634
  
```

FIG. 19V

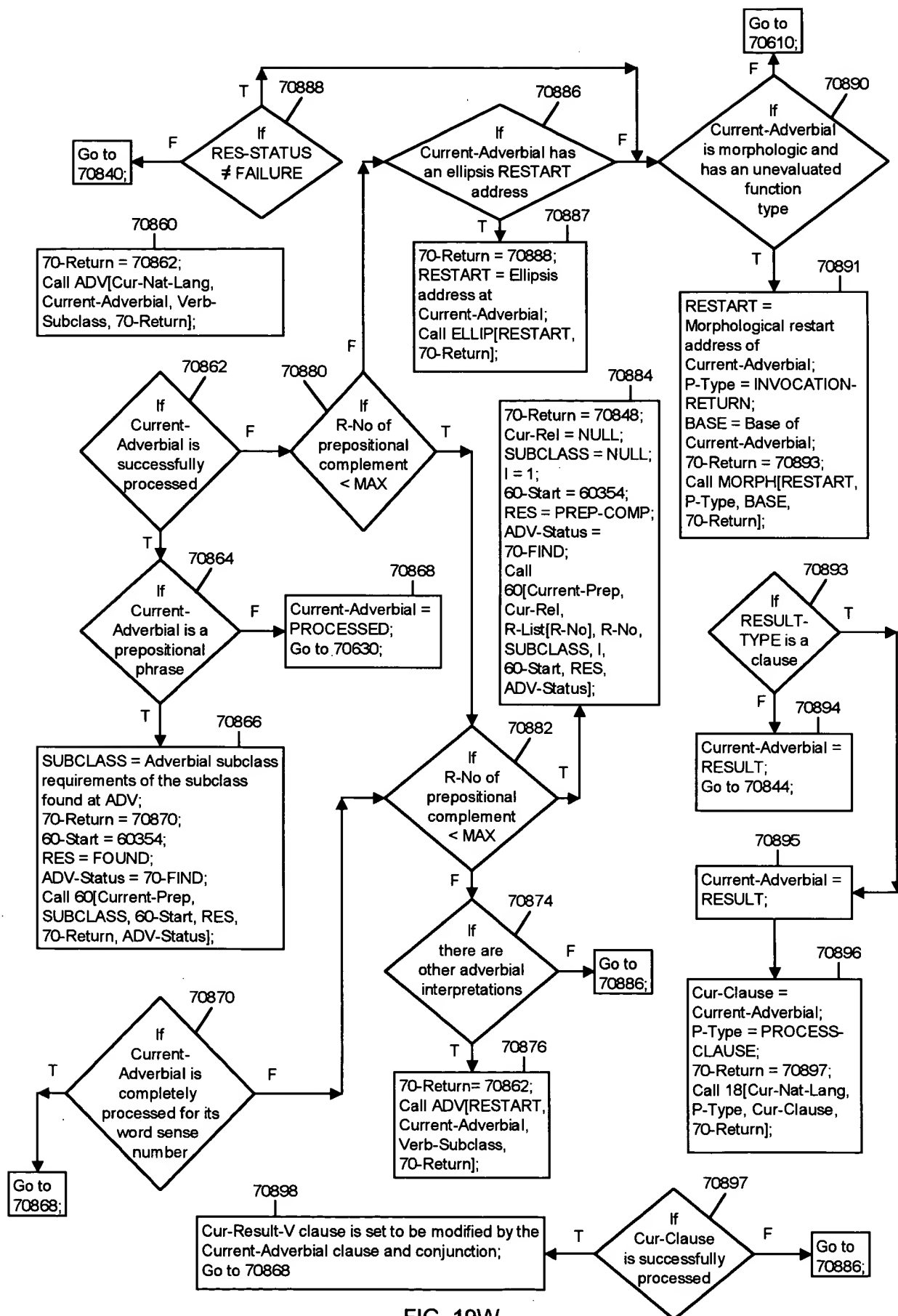


FIG. 19W

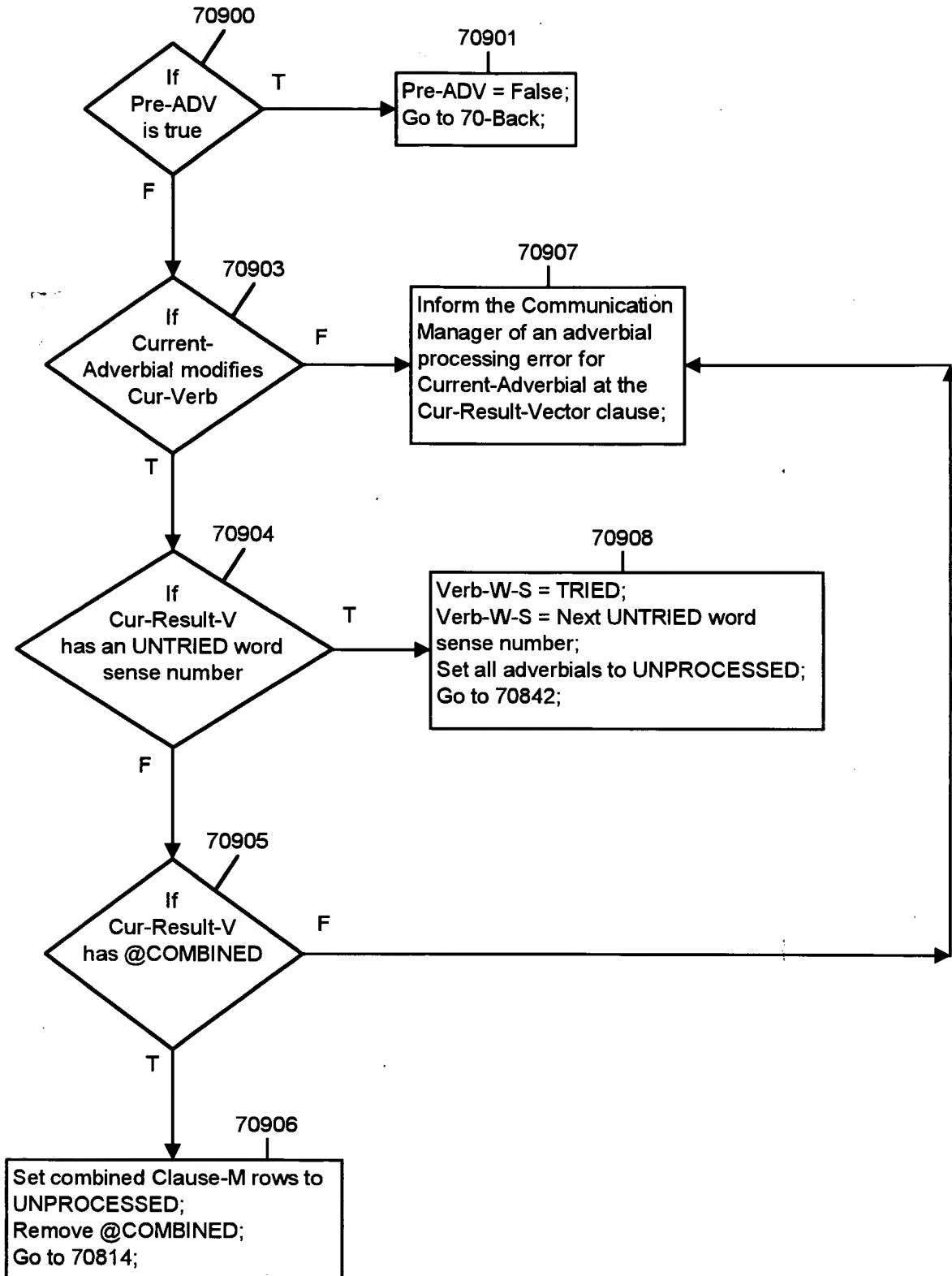


FIG. 19X

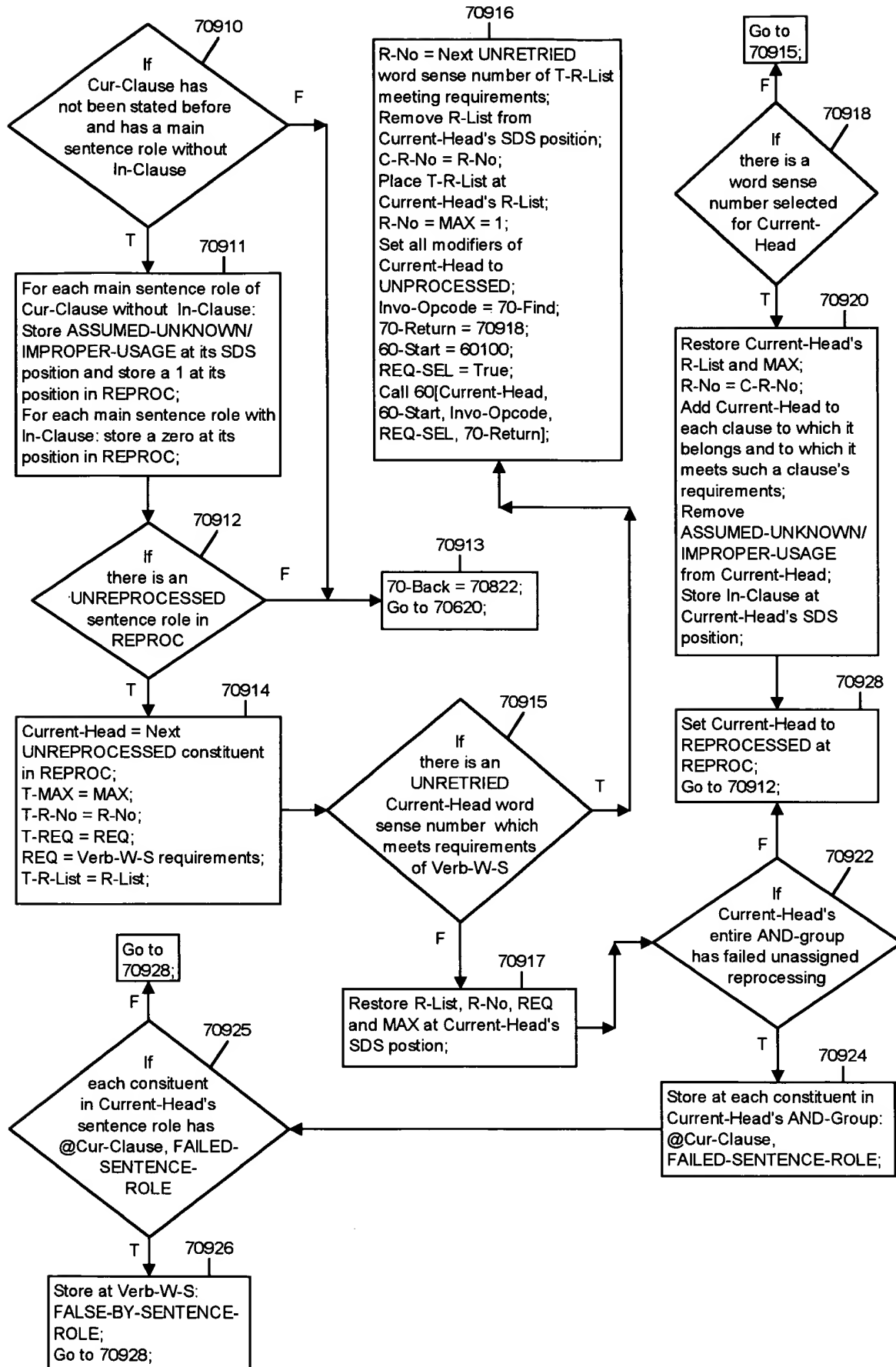


FIG.19Y

```

graph TD
    70932[70932  
Cur-Verb = 1st verb of the  
sentence with such a word sense  
number;  
Cur-Verb-W-S = 1st word sense  
number without a FALSE-BY-  
SENTENCE-ROLE truth value;  
Cur-Clause = Clause of  
Cur-Verb-W-S;] --> 70930{70930  
If  
sentence has a  
verb word sense  
number without a  
FALSE-BY-SENTENCE-ROLE  
truth value and if  
every constituent has  
In-Clause}
    70930 -- T --> 70933{70933  
If  
Cur-Clause has  
been stated before  
with the same modal  
and mood}
    70930 -- F --> 70931[70931  
C-Success = False;  
Return to caller;]
    70933 -- T --> 70934[70934  
Mark and store a pointer at the  
SDS position of Cur-Verb-W-S to  
the verb of the stated clause in  
120;  
Go to 70978;]
    70933 -- F --> 70935{70935  
If  
Cur-Verb-W-S  
requires an indirect  
object which is ellipted  
in Cur-Clause}
    70935 -- T --> 70936{70936  
If  
120 has  
Cur-Verb-W-S in a  
preceding  
clause}
    70935 -- F --> 70942[70942  
Store at indirect  
object's SDS  
position: @-V-W-S-  
Detected-Ellipsis;]
    70936 -- T --> 70937{70937  
If  
such a  
preceding clause  
has an indirect object  
different than the  
subject(s)}
    70936 -- F --> 70938[70938  
Assign indirect object  
of Cur-Clause as the  
indefinite pronoun with  
general reference;]
    70937 -- T --> 70940[70940  
Assign indirect object  
of Cur-Clause as the  
most recent, different  
indirect object in 120;]
    70937 -- F --> 70940
    70938 --> 70942
    70940 --> 70944{70944  
If  
Cur-Verb-W-S  
has an unprocessed  
modal verb}
    70942 --> 70944
    70944 -- T --> 70946[70946  
Cur-Modal = Modal verb;  
70-Return = 70948;  
Call MODAL[Cur-Nat-Lang,  
Cur-Modal, 70-Return];]
    70944 -- F --> 70949[70949  
Store result type at  
the SDS position of  
Cur-Verb-W-S;  
Go to 70950;]
    70946 --> 70948[70948  
Set a pointer at Cur-Verb-W-S  
to Cur-Modal's truth value;  
Set Cur-Modal to processed;]
    70948 --> 70944

```

FIG. 19Z

```

graph TD
    70950{70950  
If  
Cur-Verb-W-S  
has the imperative  
mood and is  
unprocessed for  
this mood}
    70952[70952  
Criteria-Set =  
Imperative-Set[Cur-Nat-Lang];  
70-Return = 70954;  
Call MODAL[Cur-Nat-Lang,  
Cur-Modal, Criteria-Set,  
70-Return];]
    70954[70954  
Store Cur-Modal's truth  
value at Cur-Verb-W-S;  
Set Cur-Verb-W-S as  
processed for the  
imperative mood;]
    70956{70956  
If  
Cur-Verb-W-S  
is in the subjunctive  
mood and is  
unprocessed for  
this mood}
    70958[70958  
Store a hypothetical truth  
value at Cur-Verb-W-S;  
Set Cur-Verb-W-S as  
processed for the  
subjunctive mood;]
    70960{70960  
If  
Cur-Clause is  
a subordinate clause  
sentence role of a verb  
implying a hypothetical  
Cur-Verb-W-S and  
is unprocessed  
for this}
    70962[70962  
Store a hypothetical truth  
value relation pointer to the  
implying verb at  
Cur-Verb-W-S;  
Set Cur-Verb-W-S as  
processed for a  
hypothetical clause  
implication;]
    70963{70963  
If  
Cur-Verb-W-S  
has its type  
selected}
    70964[70964  
Determine if Cur-Clause  
and/or the context has  
stated adverbial  
subclasses which select  
processes of  
Cur-Verb-W-S;]
    70966{70966  
If  
processes  
were  
selected}
    70968[70968  
Process-Set =  
All processes;]
    70970[70970  
Process-Set =  
Selected processes;]
    70972[70972  
Determine the processes in  
Process-Set for which all main  
sentence role constituents and  
adverbials of Cur-Clause meet  
their requirements;  
Remove processes which have  
unmet requirements;]
    70974[70974  
Search the processes in  
Process-Set for: the ALL-MAIN-  
SENTENCE-ROLE/ADVERBIALS  
match, SUBJECT match,  
INDIRECT-OBJECT match,  
DIRECT-OBJECT match,  
INDIRECT-OBJECT-AND  
DIRECT-OBJECT match, and  
ADVERBIAL-SET match  
depending upon the contents of  
PREF-PROC-V and the presence  
of direct and indirect objects in  
Cur-Clause;  
Store NULL for no matches or  
the entry number for each  
matched role type at  
respectively: ALL-M, S-M, IO-M,  
DO-M, IAD-M, and AS-M;  
Cur-Process = 1st entry stored in  
the preceding step or the  
TYPICAL-PROCESS if none was  
stored;  
Mark and store at  
Cur-Verb-W-S's SDS position:  
ALL-M, S-M, IO-M, DO-M, IAD-M,  
AS-M, and Cur-Process;  
Go to 70976;]
    70976[70976  
Go to 70976;]

    70950 -- T --> 70952
    70950 -- F --> 70956
    70952 --> 70954
    70954 --> 70974
    70956 -- T --> 70958
    70956 -- F --> 70960
    70958 --> 70974
    70960 -- T --> 70962
    70960 -- F --> 70963
    70962 --> 70974
    70963 -- T --> 70976
    70963 -- F --> 70964
    70964 --> 70966
    70966 -- F --> 70968
    70966 -- T --> 70970
    70968 --> 70972
    70970 --> 70972
    70972 --> 70974

```

FIG. 19AA

```

graph TD
    70976{70976  
If  
Cur-Clause has  
a multiple constituent  
sentence role}
    70977[70977  
Evaluate J/S-Criteria-Set for  
Cur-Process;  
Store J/S-Result-V at  
Cur-Verb-W-S's SDS position;]
    70978{70978  
If  
Cur-Verb has  
an UNPROCESSED  
clause which makes  
70930 true}
    70979[70979  
Cur-Verb-W-S = Verb word sense number  
of next clause of Cur-Verb which is  
UNPROCESSED for process selection  
meeting 70930;  
Cur-Clause = Cur-Verb-W-S's clause;  
Go to 70933;]
    70980{70980  
If  
sentence has  
an UNPROCESSED  
clause and verb word  
sense number which  
makes 70930  
true}
    70982[70982  
Cur-Verb = Next such an  
UNPROCESSED verb;]
    70984[70984  
C-Success = True;  
Ellip-Comb-Clause = False;  
REQ-SEL = False;  
Return to caller;]

    70976 -- T --> 70977
    70977 --> 70978
    70976 -- F --> 70978
    70978 -- T --> 70979
    70978 -- F --> 70980
    70980 -- T --> 70982
    70982 --> 70979
    70980 -- F --> 70984
  
```

FIG. 19BB

STATE ABSTRACT NOUN OR
ADJECTIVE WORD SENSE NUMBER FORMAT:

Word Sense Identification Number Components:

State number composed of:
 Class number, Member number;
 Owner word sense identification number;
 Value or value range;

Owner Word Sense Number:

General to specific noun word sense number;

FIG. 20A

TYPE	WORD SENSE NUMBER	VERB SETTING STATE	ADVERBIAL SUBCLASSES	PURPOSE POINTER
<u>A</u>	Identification number, most general owner word sense number	Set of verb word sense numbers setting the entry's state value	List and/or pointer to set of adjective adverbial subclasses	Pointer to related purposes
<u>B</u>	Same identification number, more specific owner word sense number of TYPE <u>A</u> entry			Pointer to related purposes

FIG. 20B

MOST GENERAL OWNER STORED WITH IDENTIFICATION NUMBER

FIG. 20C

062180-090


```

graph TD
    50100{50100  
If  
the invocation  
adjective has an unprocessed  
adverbial modifier}
    50102[50102  
Invo-Op = ADV-MOD-COMP;  
ADJ = Invocation adjective SDS position;  
50-Return = 50106;  
ADJ-W-S-Set = Adjective word sense  
numbers in the set of invocation word  
sense numbers;  
Call 50[Invo-Op, ADJ, ADJ-W-S-Set,  
50-Return];]
    50104[50104  
ADJ-W-S-Set = Adjective  
word sense numbers in the  
set of invocation word sense  
numbers;]
    50110{50110  
If  
LOOK-UP-  
Type is VERB-  
RESULT}
    50112[50112  
Store each entry of a word sense number  
in ADJ-W-S-Set compatible to the  
invocation owner and containing a verb  
setting the entry's state word sense  
number in Verb-Result-Set;  
Return Verb-Result-Set to the caller;]
    50114[50114  
Store the nearest owner entry of each  
ADJ-W-S-Set word sense number entry  
compatible with the invocation owner and  
with a purpose pointer in ADJ-Purpose-Set;  
Return ADJ-Purpose-Set to the caller;]
    50106{50106  
If  
ADV-Modifier-  
Success is true}
    50108[50108  
Return ADVERBIAL-  
MODIFIER-FAILURE  
to the caller;]

    50100 -- T --> 50102
    50100 -- F --> 50104
    50104 --> 50110
    50110 -- T --> 50112
    50110 -- F --> 50114
    50114 --> 50106
    50106 -- T --> 50110
    50106 -- F --> 50108

```

FIG. 20E

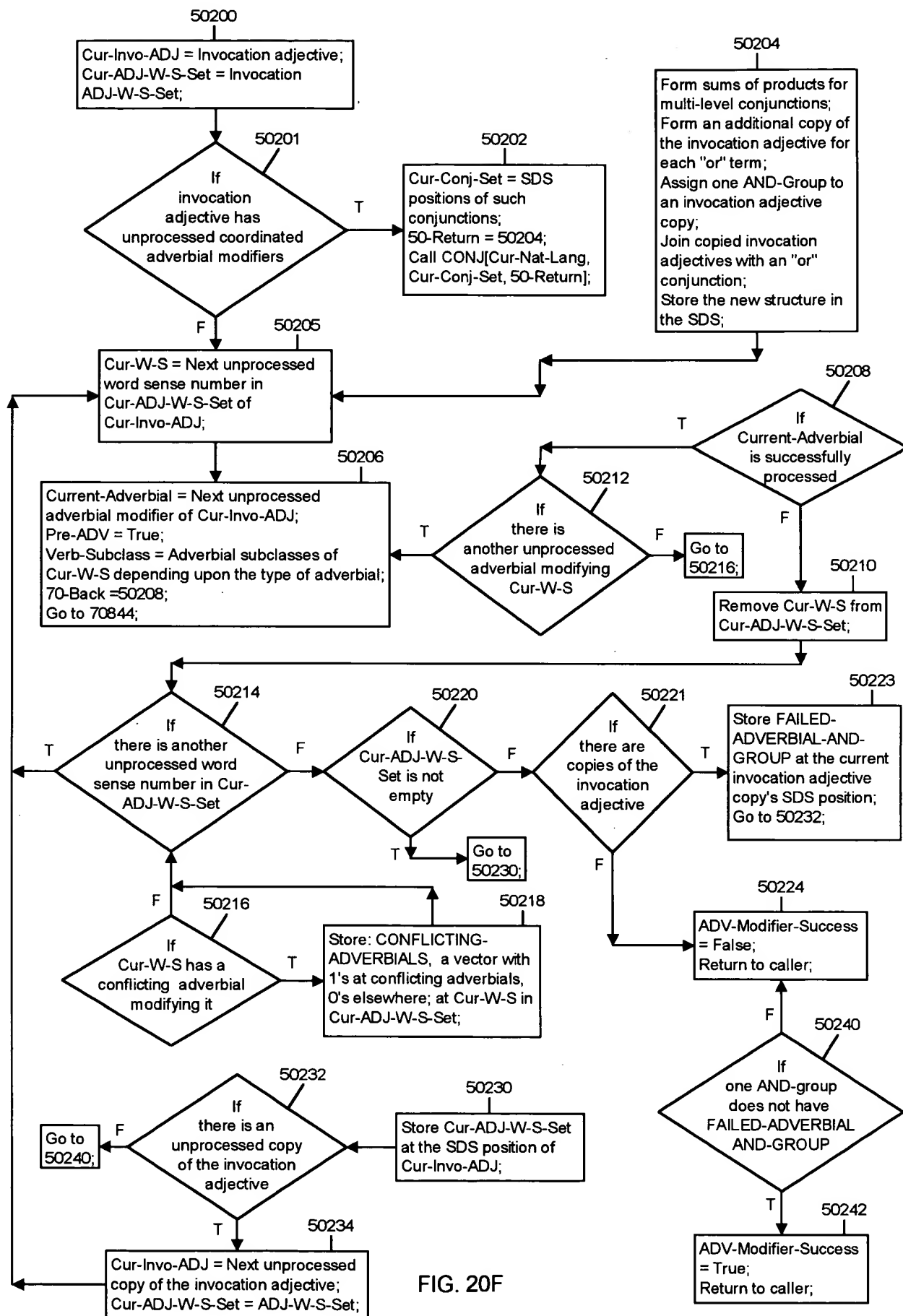


FIG. 20F

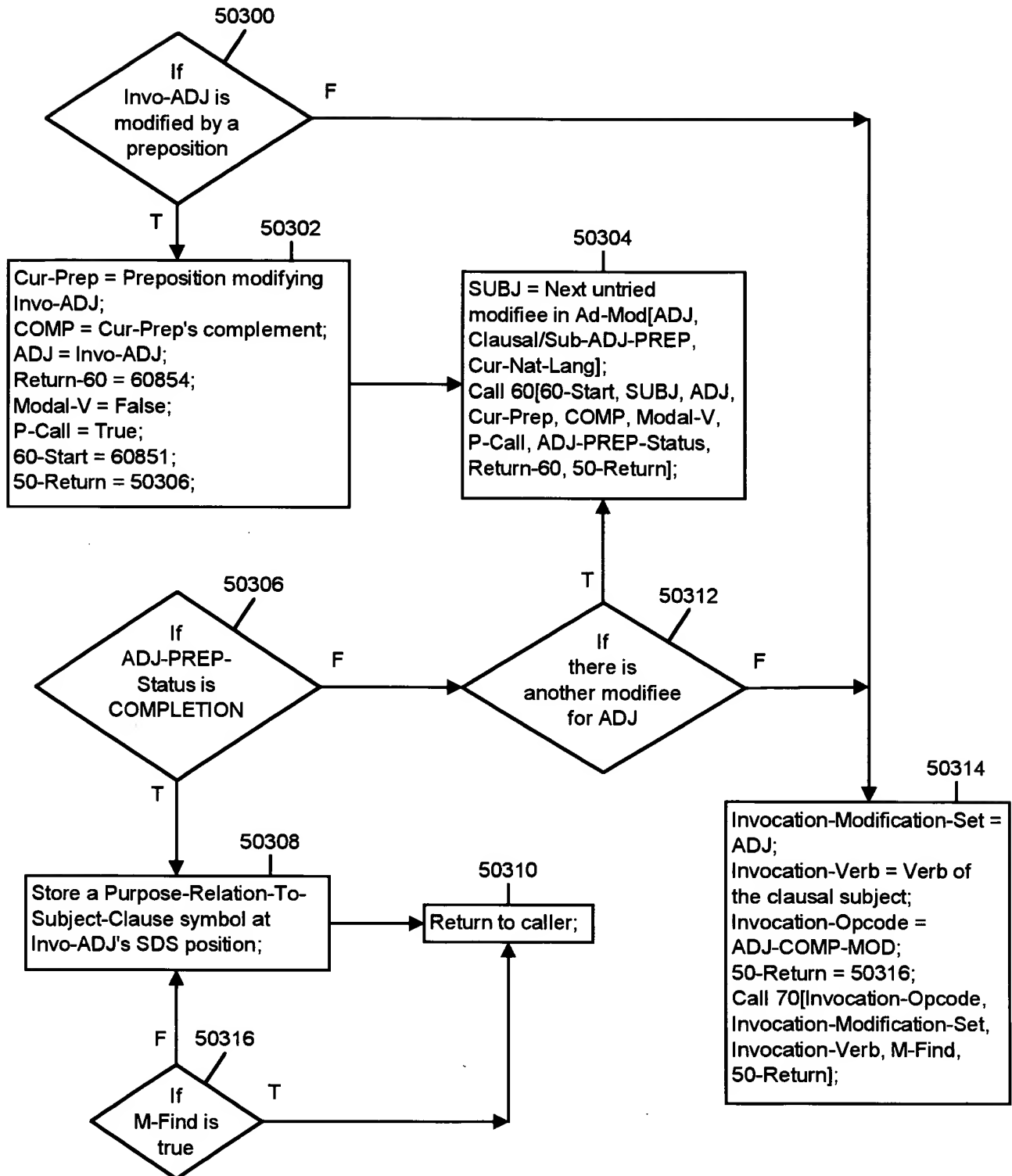


FIG. 20G

```

graph TD
    50500{50500  
If  
there is an  
unprocessed state in  
Invo-State-Set}
    50502[50502  
Return to caller;]
    50504[50504  
Cur-State = Next unprocessed  
state in Invo-State-Set;]
    50506{50506  
If  
Cur-State has  
adverbial modifiers  
with delayed  
evaluation  
functions}
    50508[50508  
Evaluate any delayed evaluation  
functions of the adverbials  
modifying the word associated  
with Cur-State;]
    50510{50510  
If  
Cur-State  
and/or its value is  
new to the  
conversation}
    50512[50512  
Store Cur-State, its value or  
value range, and a pointer to  
Cur-State's nearest owner in 80  
at New-State-Set;]
    50514[50514  
Store Cur-State, its value or  
value range, and a pointer to  
Cur-State's entry in 120 at  
Found-State-Set;]

    50500 -- F --> 50502
    50500 -- T --> 50504
    50504 --> 50506
    50506 -- T --> 50508
    50506 -- F --> 50510
    50508 --> 50510
    50510 -- T --> 50512
    50510 -- F --> 50514
    50512 --> 50500
    50514 --> 50500

```

FIG. 20H

09671580-096700

PURPOSE ADDRESS FORMAT:
<u>Purpose Identification Number:</u> Location of related purpose node table; Function number;
<u>Path Type Number:</u>
<u>Path Specificity Number:</u>
<u>Path Experience Number Set:</u>

FIG. 21A

PURPOSE NODE ENTRY FORMAT:		
OWNER'S WORD SENSE NUMBER	PURPOSE ADDRESS FUNCTION INDEX AND INTRA-FUNCTION RELATIVE FREQUENCIES	VERB WORD SENSE NUMBER OWNER'S CLAUSE CONSTITUENT INITIAL STATES
State adjective, abstract noun or verb word sense number related to the purposes of the entry;	Purpose address with pointer to the purpose realization entry in 110 or 130 and with a next memory link entry number; partitioned by function types; for each function type, each pointer has a relative frequency for its occurrence within a function;	The initial state value or value range for a non-typical state value of a clause constituent;

050700Z JUL 68

PURPOSE REALIZATION ENTRY FORMAT:					
PURPOSE NODE ENTRY ADDRESS(ES)	PROCESS APPLICATION VECTOR or NUMBER	CONSEQUENCE PURPOSE ADDRESS	MOTIVATION PURPOSE ADDRESS	OTHER ADDRESSES	ENTRY'S MEMORY LOCATION
Entry(s) of the owning purpose node address(es) plus optional path specificity vector(s);	Process application vector for purpose addresses with multiple possible processes or a process number otherwise;	Purpose or purpose tree of consequences related to the owner clause and this purpose realization	Motivation purpose or purpose of owner clause related to this purpose realization;	Addreses to purposes or purpose trees of the owner clause related to this realization such as: advantages, disadvantages, comments, alternatives, qualities, etc.; and addreses to related state representations;	Location in Memory 150 for entry's purpose realization with the owner clause

FIG. 21C

MEMORY 150 ENTRY FORMAT:							
PURPOSE REALIZATION ENTRY ADDRESS	LINK TYPES	LEVEL NUMBER AND ORDER TYPE	PRECEDING LINK ENTRY ADDRESSES	CONCURRENT LINK ENTRY ADDRESSES	SUCCEEDING LINK ENTRY ADDRESSES	RELATIVE FREQUENCY OF SEQUENTIALLY RELATED LINK ENTRIES	CONCURRENT ENTRIES SYNCHRONIZATION TYPE
Owing purpose's 110 or 130 purpose realization entry address;	# of preceding link entry addresses, # of concurrent link entry addresses, # of succeeding link entry addresses;	Number of entries between the word sense number and the start of the purpose; order type: SET, VARIABLE;	Location(s) of preceding entries in 150 of this entry including its specific address;	Location(s) of concurrent entries in 150 of this entry and access conditions;	Location(s) of succeeding entries in 150 of this entry and access conditions;	Relative frequency of a succeeding link entry relative to all its alternative succeeding link entries;	NULL - No synchronization; Number - Time units to start of concurrent clause; Range - Synchronized within lower limit to upper limit; negative limit implies number of preceding units; positive limit implies number of succeeding units;

FIG. 21D

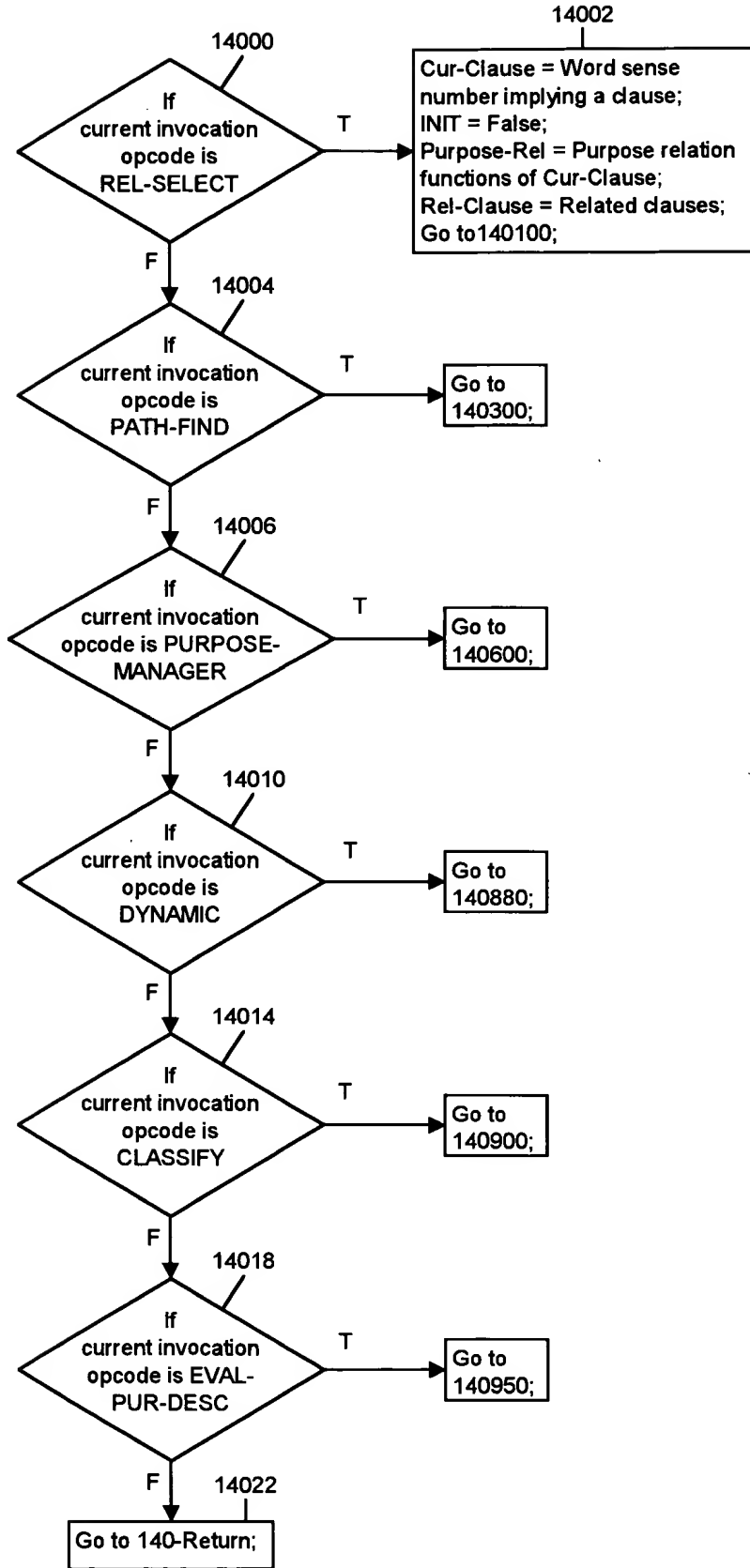


FIG. 21E

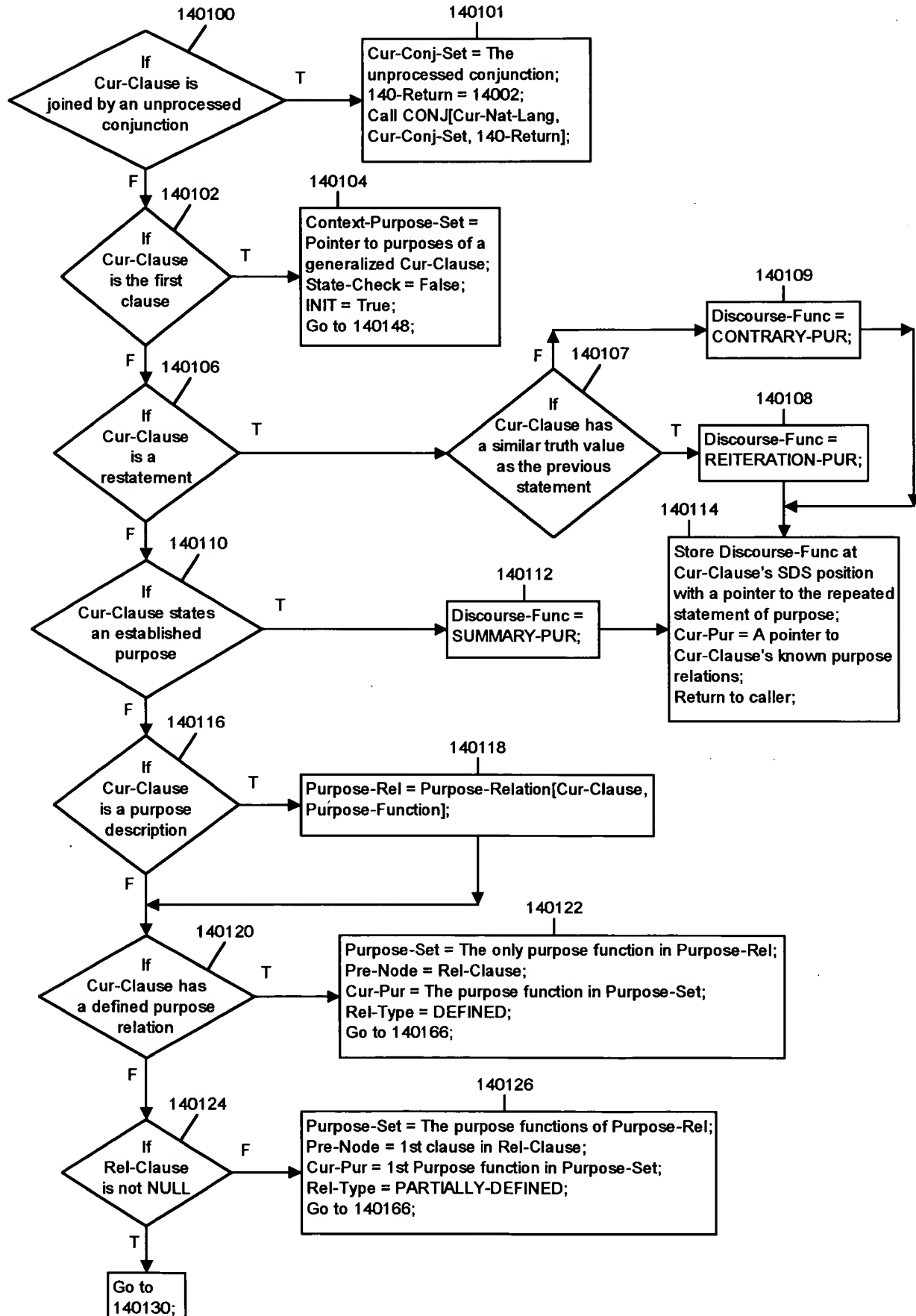


FIG. 21F

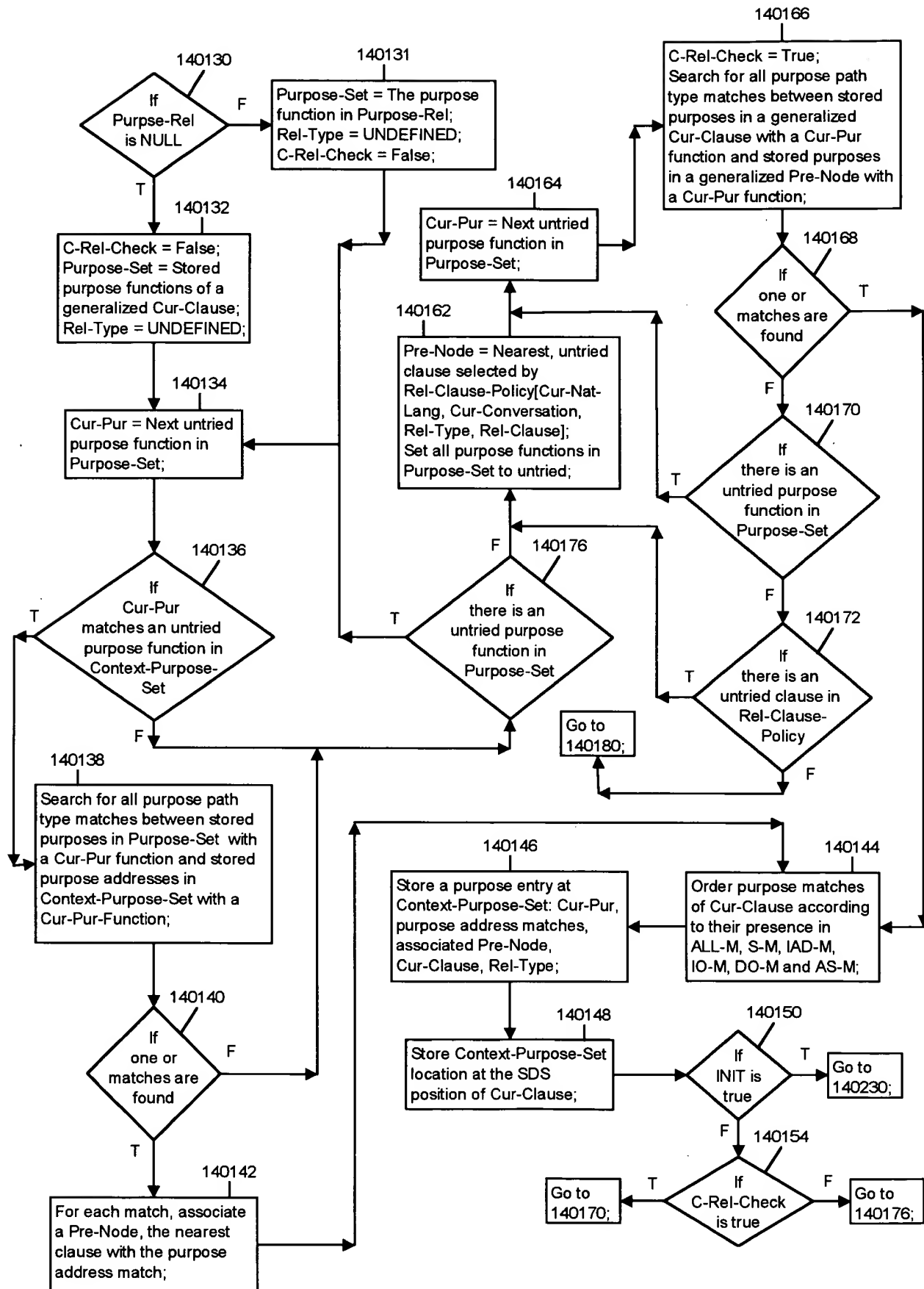


FIG. 21G

```

graph TD
    140180[Go to Pur-Class-Set[Cur-Nat-Lang, Cur-App];] --> 140182[140-Return = 140190; RS = False; Check-Clause = Cur-Clause;]
    140182 --> 140183{If Purpose-Set has a MOTIVATION function and a stored function was not found}
    140183 -- T --> 140184[140-Restart = 140186; CLASS = MOTIVATION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140183 -- F --> 140186{If Purpose-Set has an EXCEPTION function and a stored function was not found}
    140186 -- T --> 140188[140-Restart = 140196; CLASS = EXCEPTION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140186 -- F --> 140196{If Purpose-Set has a INFORMATION function and a stored function was not found}
    140196 -- T --> 140198[140-Restart = 140200; CLASS = INFORMATION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140196 -- F --> 140200{If Purpose-Set has a CONDITION function and a stored function was not found}
    140200 -- T --> 140202[140-Restart = 140204; CLASS = CONDITION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140200 -- F --> 140204{If Purpose-Set has a LISTING function and a stored function was not found}
    140204 -- T --> 140206[140-Restart = 140208; CLASS = LISTING; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140204 -- F --> 140208{If Purpose-Set has a CONTRAST function and a stored function was not found}
    140208 -- T --> 140210[140-Restart = 140212; CLASS = CONTRAST; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140208 -- F --> 140190{If a process was classified}
    140190 -- T --> 140192[Store a purpose entry at Context-Purpose-Set: CLASS, NULL, related clause, Cur-Clause, C-Pur; Store entry location at the SDS position of Cur-Clause;]
    140190 -- F --> 140194[Go to 140-Restart;]
    140194 --> 140212{If Purpose-Set has a ALTERNATIVE/PREFERENCE function and a stored function was not found}
    140212 -- T --> 140214[140-Restart = 140216; CLASS = ALTERNATIVE/PREFERENCE; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140212 -- F --> 140216{If Purpose-Set has a PROPORTION function and a stored function was not found}
    140216 -- T --> 140218[140-Restart = 140220; CLASS = PROPORTION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140216 -- F --> 140220{If Purpose-Set has a CONCESSION function and a stored function was not found}
    140220 -- T --> 140222[140-Restart = 140230; CLASS = CONCESSION; Call 140[CLASSIFY, CLASS, RS, Check-Clause, 140-Return];]
    140220 -- F --> 140230[Go to 140230;]
  
```

FIG. 21H

FIG. 21H


```

graph TD
    140230{140230  
If a  
purpose relation  
was found or INIT  
is true} -- T --> 140254[140254  
Store at Context-Purpose-Set:  
DEFAULT-DESCRIPTION,  
NULL, NULL, Cur-Clause,  
Rel-Type;]
    140230 -- F --> 140250{140250  
If  
Cur-Clause's  
relation was  
searched for with an  
AMBIGUOUS  
conjunction}
    140250 -- T --> 140252[140252  
Cur-Clause = Alternative  
conjunction;  
Set the conjunction to  
UNAMBIGUOUS;  
Go to 14002;]
    140250 -- F --> 140248{140248  
If  
State-Check  
is true}
    140248 -- T --> 140260[140260  
State-Check-Set = States  
and positions of the  
constituents of Cur-Clause  
which match states in  
State-Select[Cur-App];  
Mark those states which  
require a process  
determination;]
    140248 -- F --> 140280[140280  
Return to  
caller;]
    140260 --> 140262{140262  
If  
there is an  
unchecked state in  
State-Check-Set}
    140262 -- T --> 140264[140264  
Cur-C-State = Next  
unchecked state;]
    140262 -- F --> 140276{140276  
If  
State-Check-  
Set is not  
empty}
    140276 -- T --> 140278[140278  
Store State-Check-Set  
at Cur-Clause's SDS  
position;]
    140276 -- F --> 140244[140244  
Store Cur-Time,  
and T-Type, at  
Cur-Clause's SDS  
position;]
    140264 --> 140266{140266  
If  
Cur-C-State  
has a different  
value than in 120  
or is new}
    140266 -- T --> 140268[140268  
Store previous and current  
Cur-C-State at the  
Cur-C-State position in  
State-Check-State;]
    140266 -- F --> 140270[140270  
Remove Cur-C-State  
from State-Check-Set;]
    140268 --> 140232{140232  
If  
Time-Check is  
true and a doer sets  
states or Cur-Clause  
has a time  
adverbial}
    140232 -- T --> 140234{140234  
If  
Cur-Clause is an  
ALL-M match or  
there is a stored  
time point in  
80 or 90}
    140234 -- T --> 140236[140236  
Cur-Time = The time  
point of Cur-Clause;  
T-Type = DEFINITE;]
    140234 -- F --> 140238{140238  
If  
there is a  
time point for the  
nearest Pre-Node  
of Cur-Clause}
    140238 -- T --> 140242[140242  
Cur-Time = The typical  
transition time for  
Cur-Clause;  
T-Type = TRANSITION;]
    140238 -- F --> 140240[140240  
Cur-Time = The time  
point of the most  
recent Pre-Node + the  
transition time;  
T-Type = RELATIVE;]
    140242 --> 140244
    140240 --> 140244
    140244 --> 140254

```

FIG. 21I

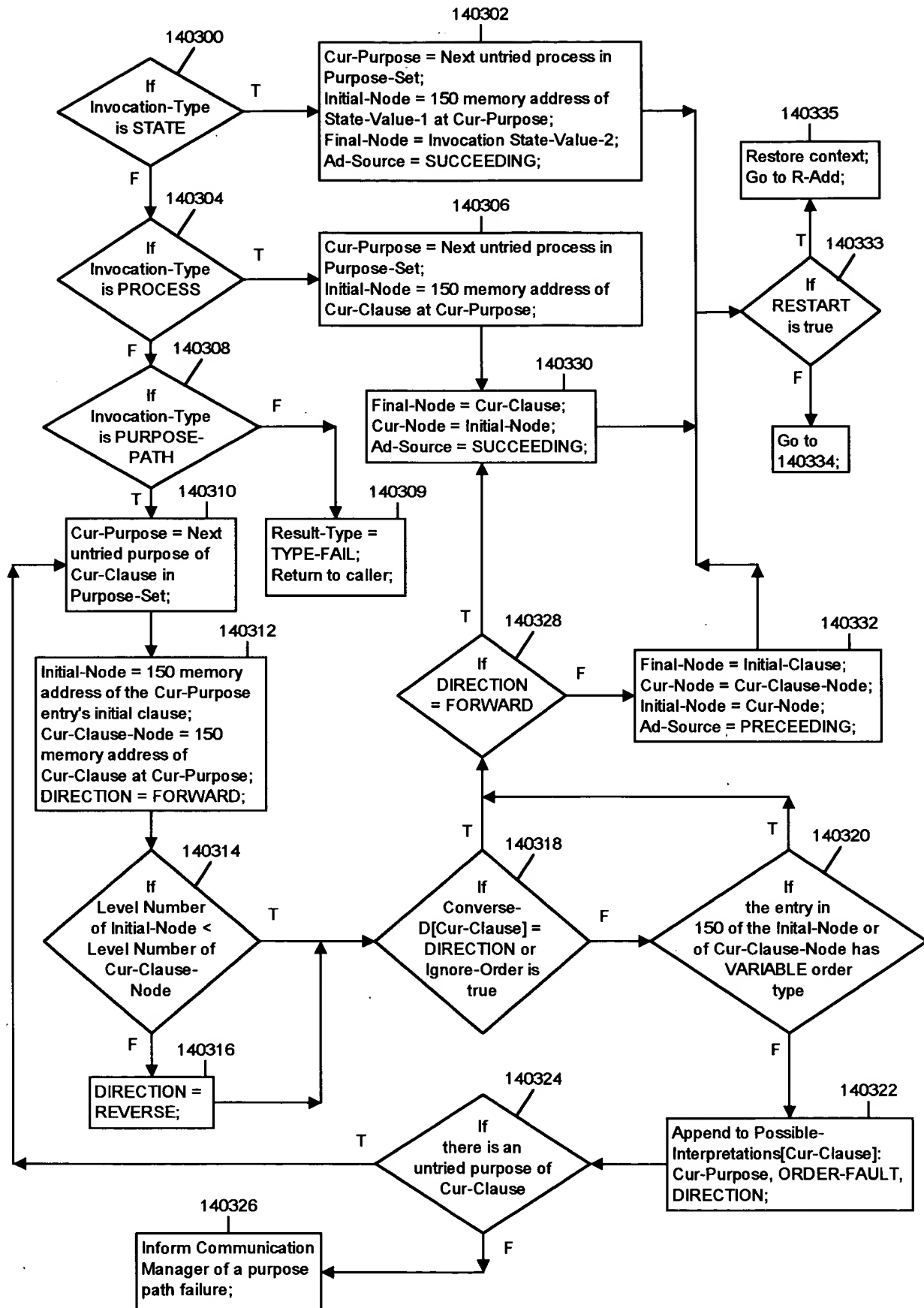


FIG. 21J

Zero: PATH, In-Path, Active-Path, BACK, C-Branch, Assumed-Sentence-Roles, Time-Vec, S-Cur-Time, Modal-Vec;
PATH[1, 1, 1] = PATH[1, 1, 2] =
Cur-Node;
BACK[1, 1] = Number of Ad-Source
addresses at Cur-Node;
O-Source = Ad-Source;
Path-No = NODE = 1;
Add 1 to In-Path and Active-Path where it
is set to PROCESSED;
Fin-Node = S-Path = False;

Next-Node = Next untried Ad-Source
address at Cur-Node which has an
access condition set met by the context;
BACK[Path-No, NODE] =
BACK[Path-No, NODE] - 1 - the number
of just failed condition sets;
Result-Type = ACCESS-CONDITION;
R-Add = 140336;

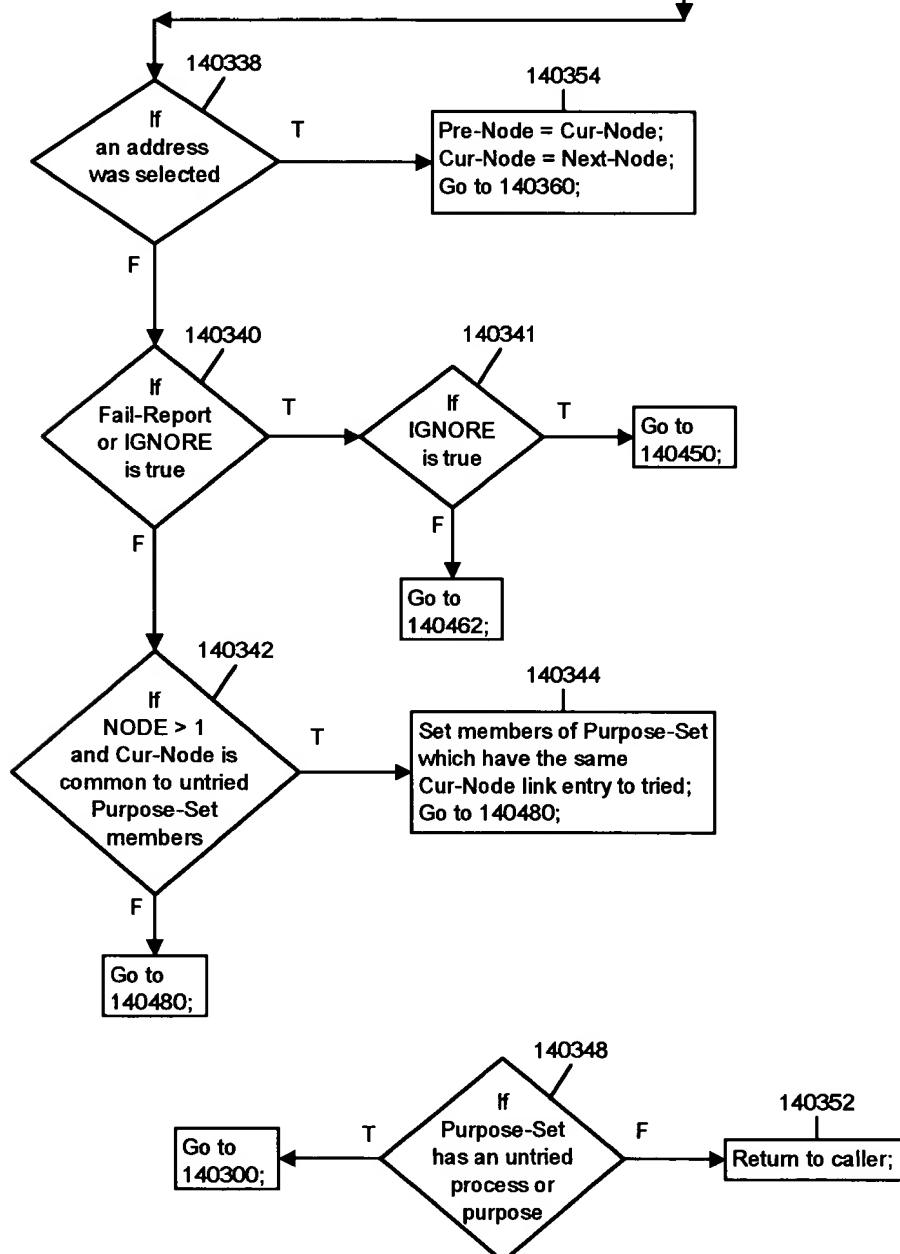
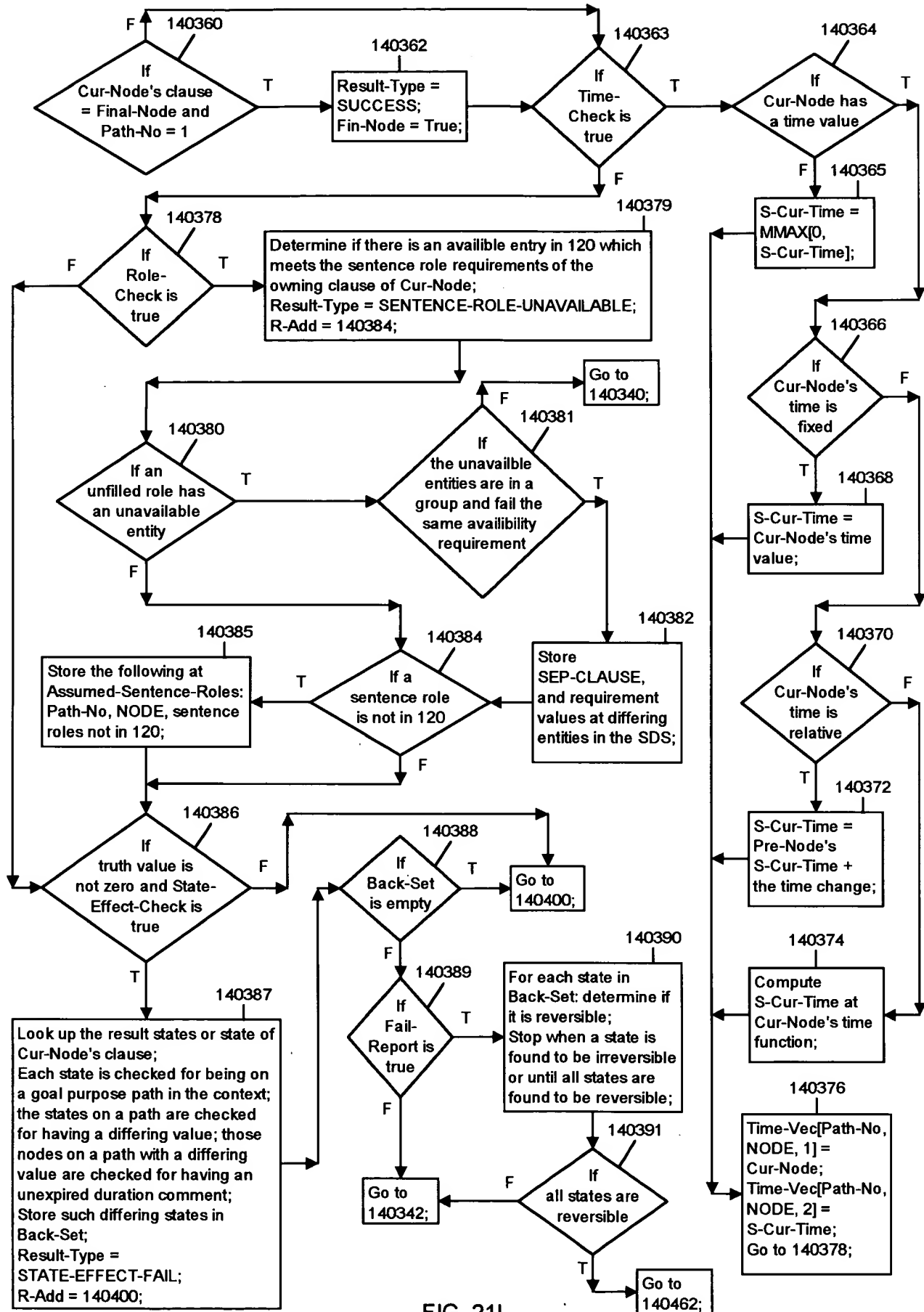


FIG. 21K



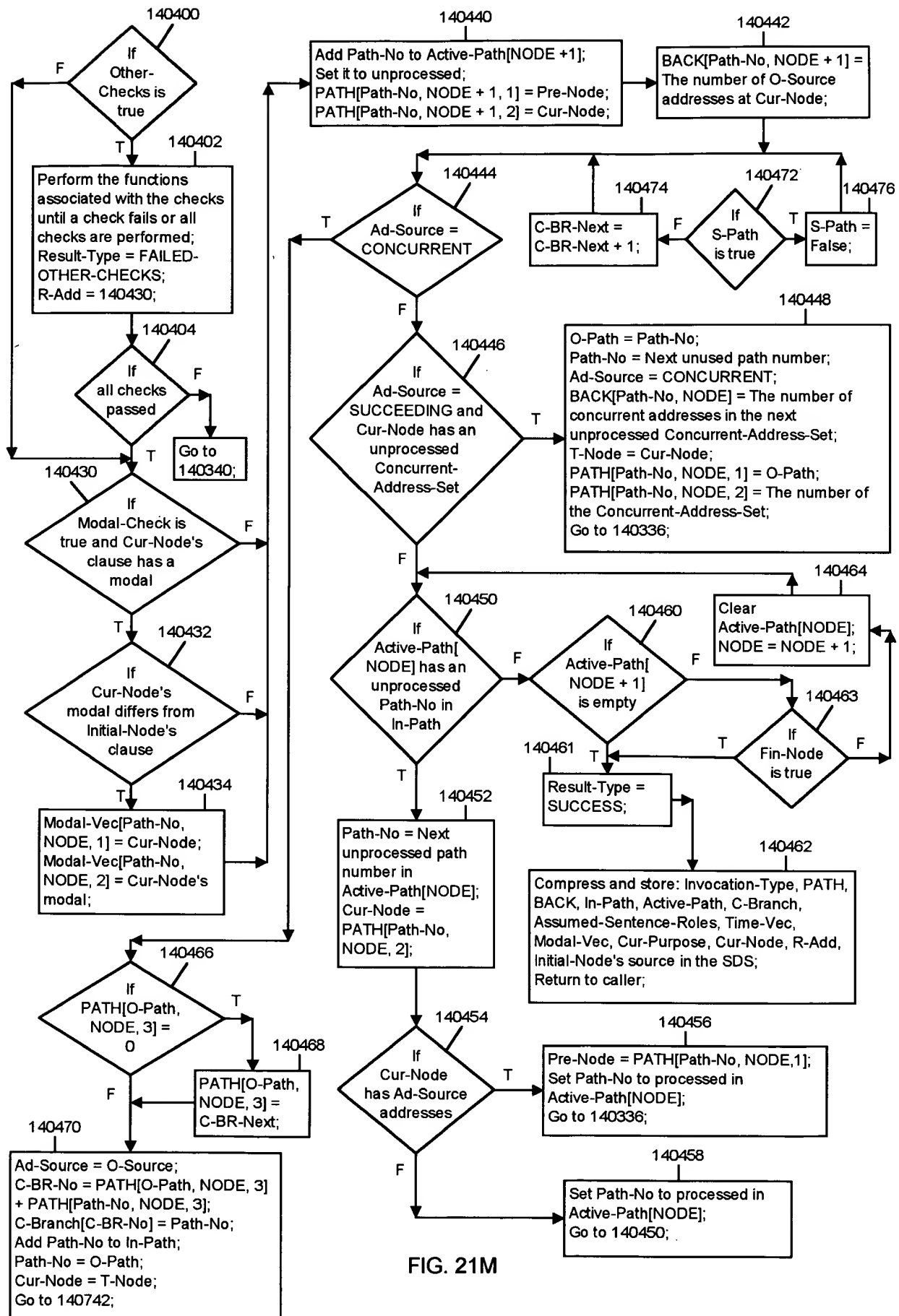


FIG. 21M

[illegible]

FIG. 21N

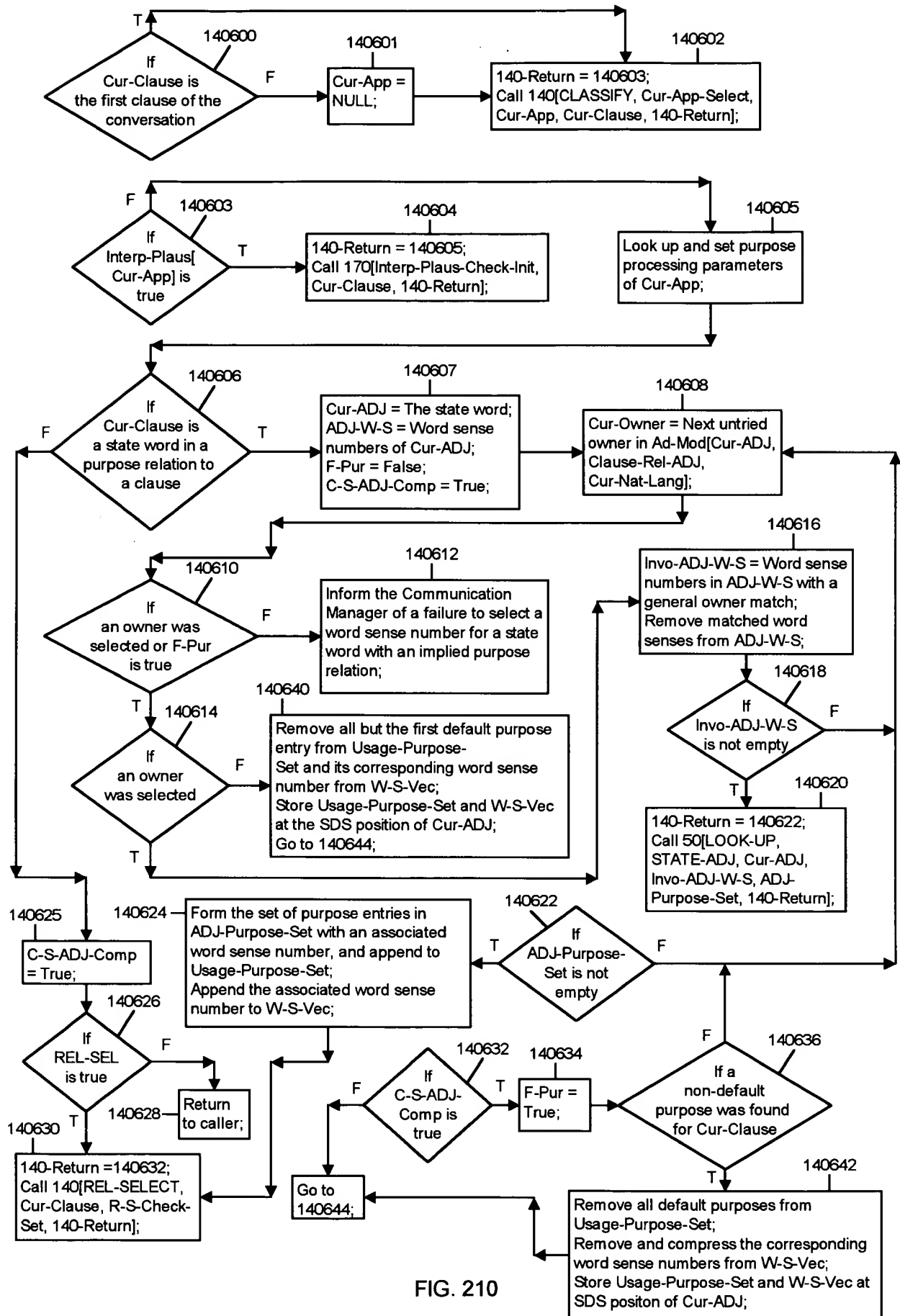


FIG. 210


```

graph TD
    140710[Cur-C-Proc = False;] --> 140711{If Process-Path-Find is true and Cur-Clause has a Process-Set}
    140711 -- T --> 140712[Purpose-Set = Union of (ALL-M, S-M, IO-M, DO-M, IAD-M, AS-M, Process-Set);  
Invocation-Type = PROCESS;  
RESTART = False;  
140-Return = 140714;  
Call 140[PATH-FIND, Invocation-Type, Purpose-Set, Cur-Clause, Proceed-Type[PROCESS], P-F-Check-Set[PROCESS], Other-Checks[PROCESS], RESTART, Result-Type, 140-Return];]
    140711 -- F --> 140714{If Result-Type = SUCCESS}
    140714 -- T --> 140722[Store path data in the SDS;  
Cur-C-Proc = True;]
    140714 -- F --> 140716{If Proceed-Type[PROCESS] = Fail-Report}
    140716 -- T --> 140718[Call Cur-App[Process-Path-Fail, Cur-Purpose, Cur-Node, Result-Type, R-Add, 140-Return];]
    140716 -- F --> 140720[Inform the Communication Manager of a path find failure in a process path realization;]
    140722 --> 140724{If Plaus-Expect-Check is true}
    140724 -- T --> 140726[140-Return = 140730;  
Call 170[PLAUS-EXP-CHECK, Cur-Clause, Cur-App, Result-Type, P-E-Vec, 140-Return];]
    140724 -- F --> 140740[Store states of stated adjectives in Cur-Clause at Invocation-State-Set;]
    140726 --> 140730{If Result-Type = SUCCESS}
    140730 -- T --> 140732[Store P-E-Vec at the SDS position of Cur-Clause's verb or clause implying adjective;]
    140730 -- F --> 140734{If Plaus-Expect-Fail is true}
    140734 -- T --> 140736[140-Return = 140732;  
Call Cur-App[Plause-Expect-Check-Fail, Cur-Clause, P-E-Vec, 140-Return];]
    140734 -- F --> 140738[Inform the Communication Manager of a plausibility and expectedness check failure;]
    140740 --> 140742{If Invocation-State-Set is not empty}
    140742 -- T --> 140744[140-Return = 140746;  
Call 50[ADJECTIVE-COMPLETION, Invocation-State-Set, 140-Return];]
    140742 -- F --> 140746{If Cur-App[Other-Checks] is true}
    140746 -- T --> 140748[140-Return = 140750;  
Call Cur-App[O-Checks, Cur-Clause, 140-Return];]
    140746 -- F --> 140750[Go to 140750;]
  
```

FIG. 21Q

FIG. 21Q

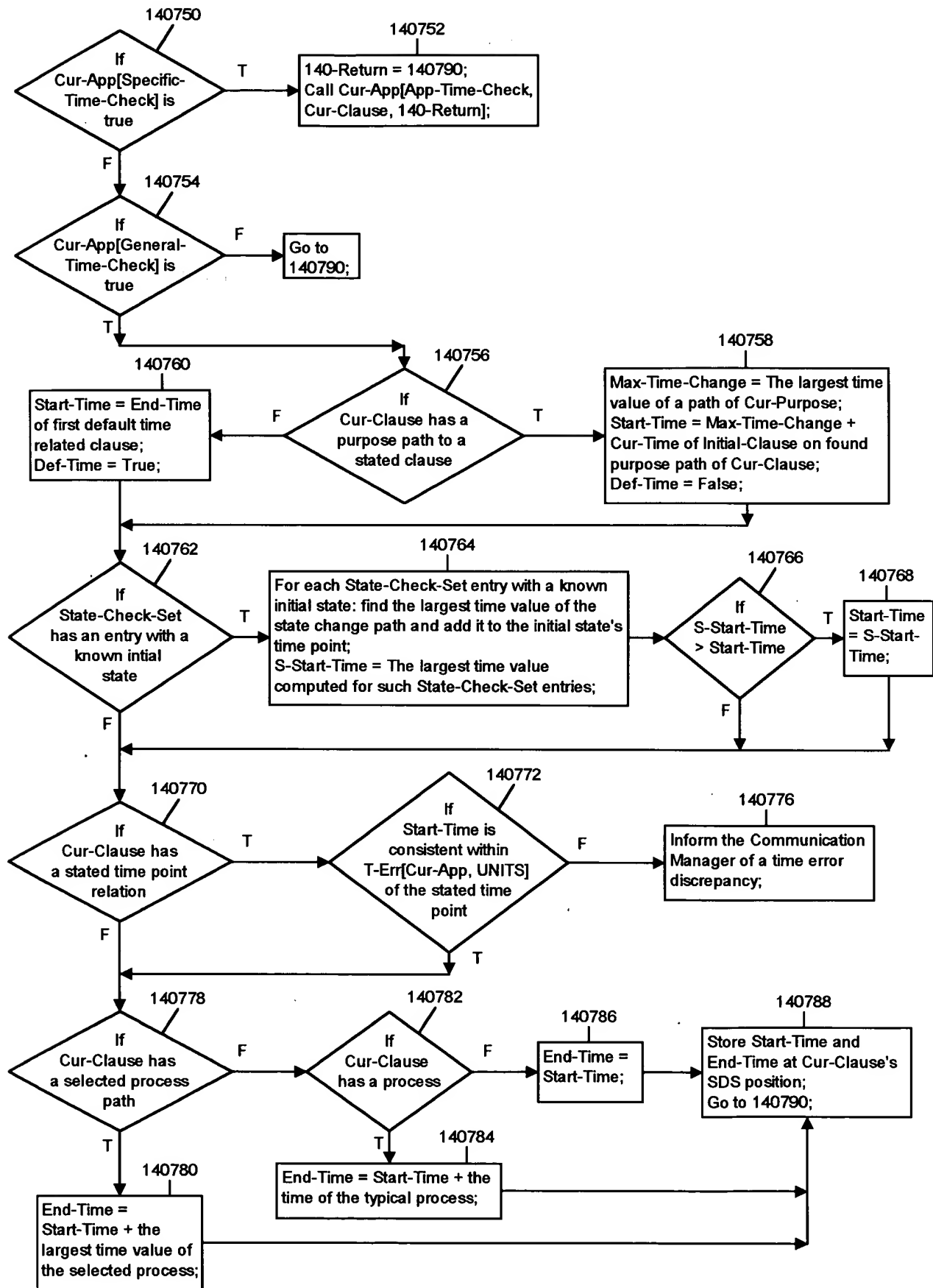


FIG. 21R

```

graph TD
    140790{140790  
If  
Modal-Func[  
Cur-App] is true} -- T --> 140792[140792  
Cur-Modal =  
Modal-Comp[Modal-Vec,  
Cur-Clause, Cur-App]:]
    140790 -- F --> 140804{140804  
If  
RESPONSE[  
Cur-App] is true}
    140804 -- F --> 140806[140806  
Return to caller;]
    140804 -- T --> 140808{140808  
If  
Pur-Rel-P-F  
is true}
    140808 -- T --> 140812{140812  
If  
Pur-Rel has a  
modification  
function}
    140812 -- T --> 140814[140814  
Purpose-Type =  
PURPOSE-MODIFICATION;]
    140812 -- F --> 140816{140816  
If  
Pur-Rel is  
an established  
purpose}
    140816 -- T --> 140820[140820  
Purpose-Type =  
NEW-PURPOSE;]
    140816 -- F --> 140818[140818  
Purpose-Type =  
PURPOSE-CONTINUATION;]
    140814 --> 140822[140822  
140-Return = 140824;  
Call Cur-App[RESPONSE,  
Cur-Clause, Unstored-Rel,  
Purpose-Type, Cur-C-Proc,  
State-Check-Set, Action-Type,  
Action-List, 140-Return];]
    140820 --> 140822
    140818 --> 140822
    140822 --> 140832{140832  
If  
Next-Clause  
is a state or  
state abstract  
noun}
    140832 -- T --> 140834[140834  
Out-List[CLAUSE-TYPE, Next-Pos] =  
ENUMERATED-STATE;  
Out-List[OWNER, Next-Pos] =  
Context, general type, stored, or  
designated owner;  
Out-List[VERB, Next-Pos] = Verb word  
sense number implied by Next-Clause's  
type;  
Subject-Number = S-Number[Cur-  
Nat-Lang, Out-List[OWNER, Next-Pos]];  
Go to 140842;]
    140832 -- F --> 140838[140838  
Go to]
    140792 --> 140794{140794  
If  
Cur-Clause  
has a stated  
modal}
    140794 -- T --> 140796[140796  
Modal-Dif =  
|Cur-Modal minus  
the stated modal  
value;]
    140794 -- F --> 140802[140802  
Store Cur-Modal  
at Cur-Clause's  
SDS position;]
    140796 --> 140798{140798  
If  
Modal-Report  
is true and  
Modal-Dif >  
Modal-Err[  
Cur-App]}
    140798 -- T --> 140800[140800  
Inform the Communication  
Manager of a modal error  
discrepancy;]
    140798 -- F --> 140802
    140802 --> 140829[140829  
Return to caller;]
    140829 --> 140824{140824  
If  
Action-Type =  
CONTINUE or  
RESPONSE-  
READY}
    140824 -- T --> 140829
    140824 -- F --> 140826{140826  
If  
Action-Type =  
STANDARD-  
OUTPUT}
    140826 -- T --> 140827[140827  
Init-Pos = Next unused  
positon in Out-List;  
Cur-Resp-Pur-Address =  
Purpose address of the  
response;  
Response-Object = Address  
of the previous input SDS;  
Cur-Resp-Func = Function  
of the response purpose;]
    140827 --> 140834
    140826 -- F --> 140828[140828  
Inform the Communication  
Manager of an output  
formation error;]
    140828 --> 140800

```

FIG. 21S

FIG. 21S

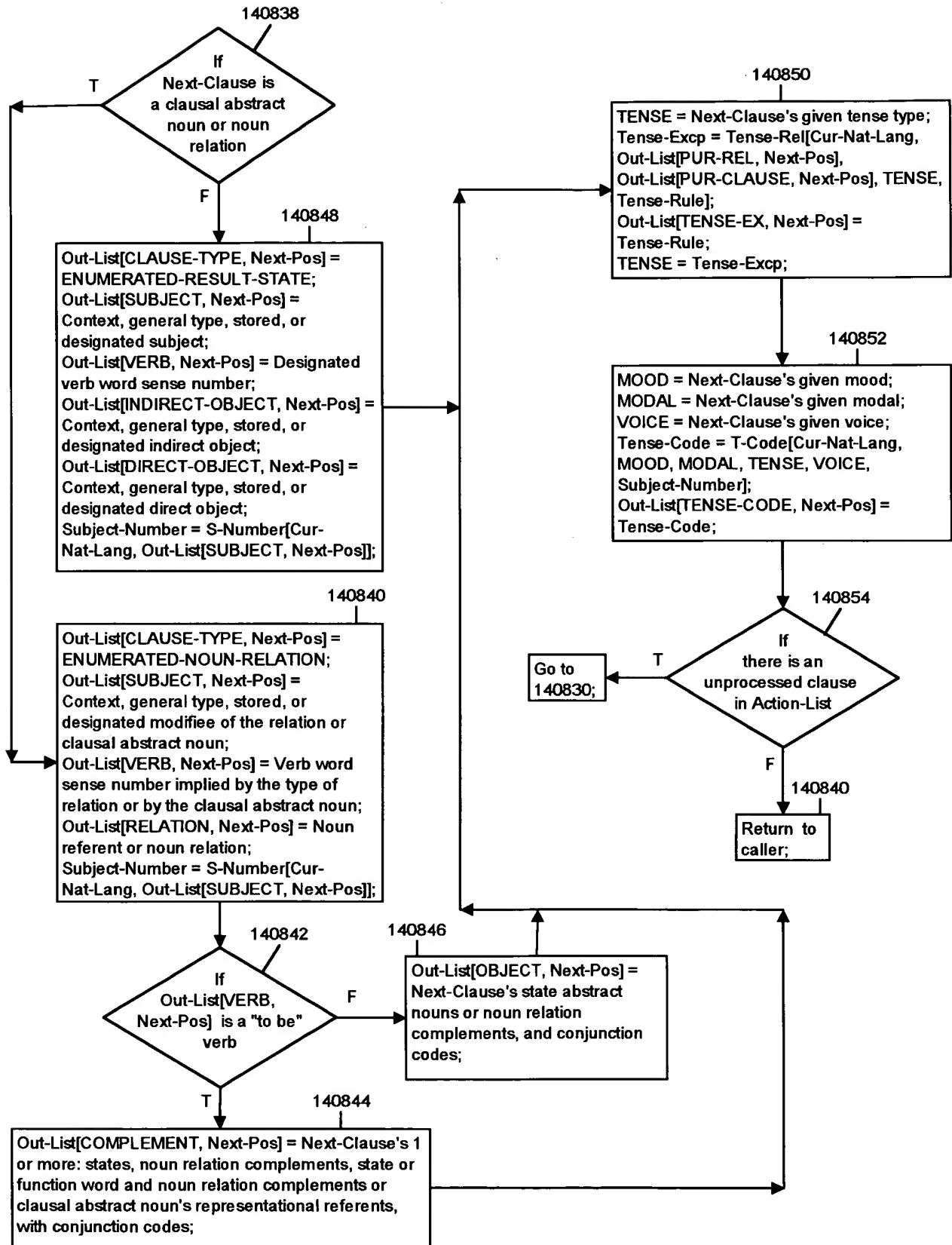


FIG. 21T

**140-Return = 140882;
Call Dy-Pur-Name[Dynamic-Purpose-Parameters, Pur-Set, W-S-No, Pro-Type, P-F-C-Set, Other-C, SDS-Tem, D-Parm, 140-Return];**

Invocation-Type = PROCESS;
Purpose-Set = Pur-Set;
Cur-Clause = W-S-No;
RESTART = Invo-RS;
140-Return = 140884;
SDS = SDS-Tem;
D-Spec-Vec = D-Param;
Call 140[PATH-FIND, Invocation-Type,
Purpose-Set, Cur-Clause, Pro-Type,
P-F-C-Set, Other-C, RESTART,
Result-Type, 140-Return];

```
Return Dy-Pur-Name[Path-Report,  
Cur-Purpose, Cur-Node,  
Result-Type, R-Add];
```

140-Return = 140902;
Call Class-Name[Invo-Obj,
Classification-Purpose-Parameters,
Pur-Set, W-S-No, Pro-Type, P-F-C-Set,
Other-C, SDS-Tem, C-Parm, 140-Return];

**Invocation-Type = PROCESS;
Purpose-Set = Pur-Set;
Cur-Clause = W-S-No;
RESTART = Invo-RS;
140-Return = 140904;
SDS = SDS-Tem;
C-Spec-Vec = C-Parm;
Check-Object = Invo-Clasiffee;
Call 140[PATH-FIND, Invocation-Type,
Purpose-Set, Cur-Clause, Pro-Type,
P-F-C-Set, Other-C, RESTART,
Result-Type, 140-Return];**

**Return Class-Name[Path-Report,
Cur-Purpose, Cur-Node,
Result-Type, R-Add];**

00/260-0257/950

140950

Transfer Invo-P-No parameters from
Invo-Table to Pur-Desc-Table;
140-Return = 140952;
Call Invo-Pur-Desc-Name[Pur-Desc-Table];

140952

Return Result-Table
to caller;

FIG. 21V

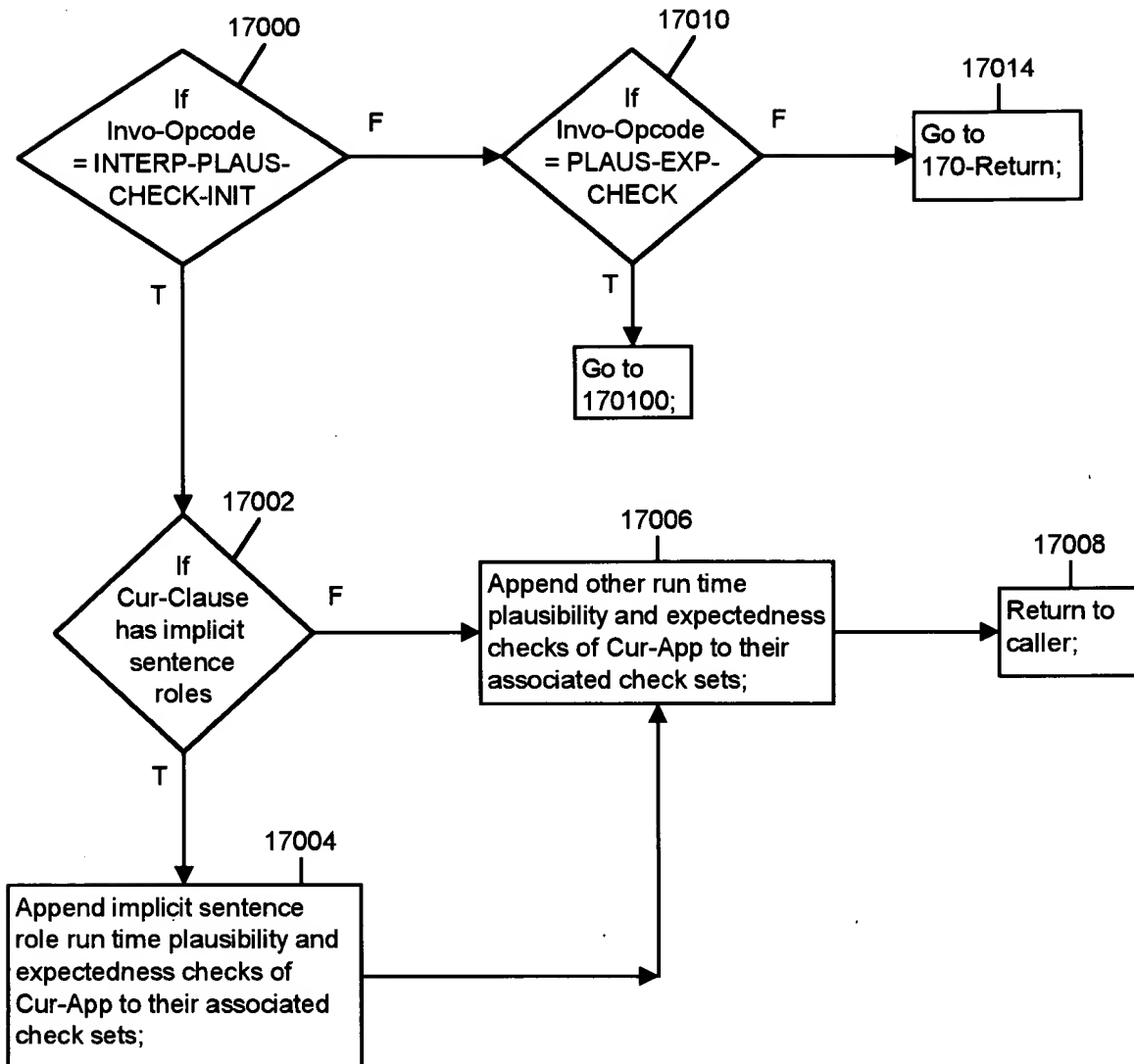


FIG. 22A

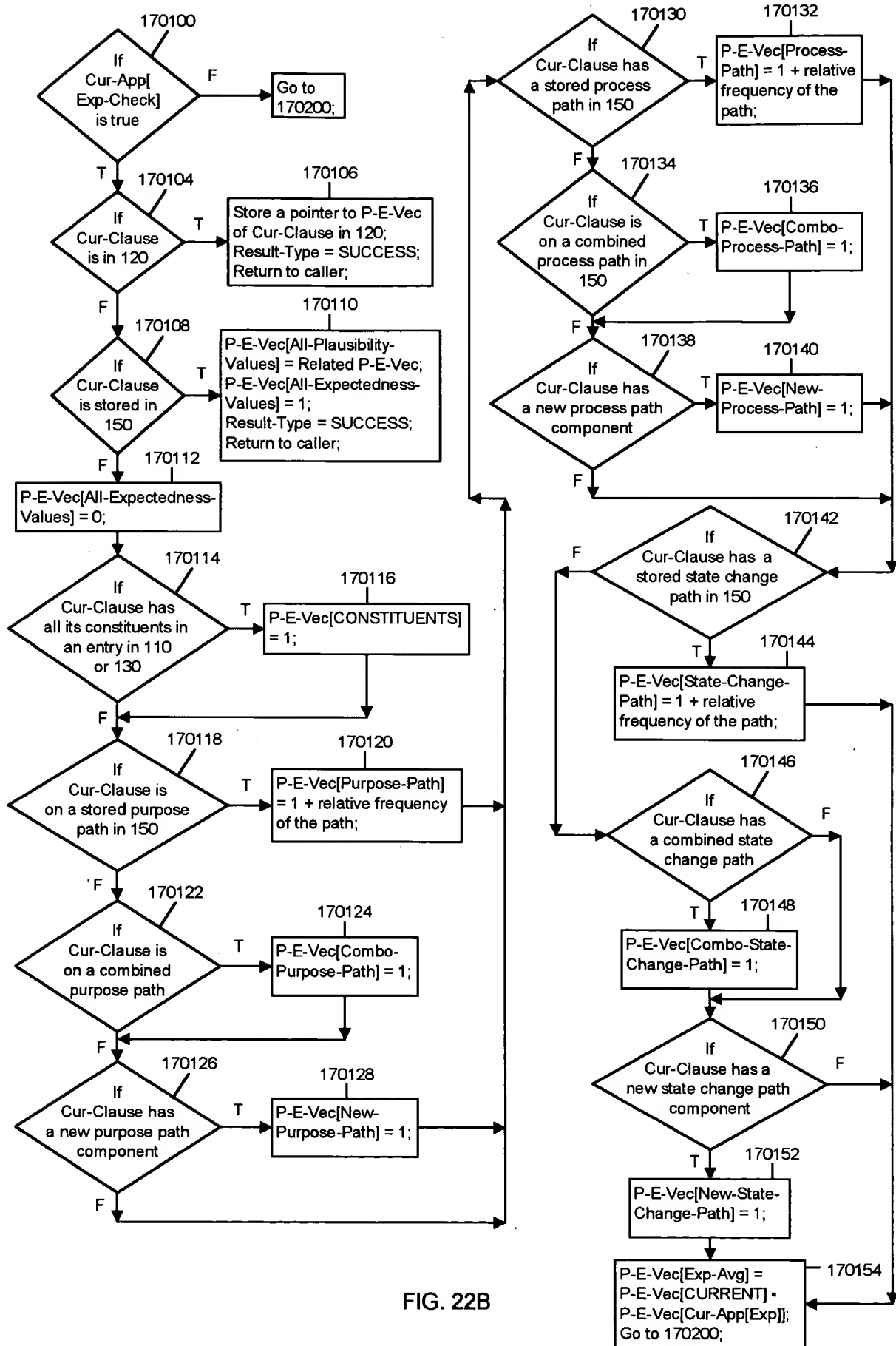


FIG. 22B


```

graph TD
    170200{If Cur-App[Plausibility-Check] is true} -- F --> 170202[Result-Type = SUCCESS; Return to caller;]
    170200 -- T --> 170204[P-E-Vec[All-Plausibility-Checks] = 0;]
    170204 --> 170206{If Cur-App[Pur-Plaus] is true}
    170206 -- F --> 170240[Go to 170240;]
    170206 -- T --> 170210{If Cur-Purpose is established or is in 150}
    170210 -- T --> 170212[P-E-Vec[Purpose-Plausibility-] = 1;]
    170212 --> 170240
    170210 -- F --> 170214{If Cur-Purpose is on an unestablished path}
    170214 -- T --> 170216{If Cur-Purpose has a benefits classification purpose}
    170216 -- T --> 170220[CLASS = Purpose-Benefits; RS = False; Invo-Obj = Cur-Purpose; 170-Return = 170222; Call 140[CLASSIFY, CLASS, RS, Invo-Obj, 170-Return];]
    170216 -- F --> 170218[P-E-Vec[Doer-Purpose-Ben] = P-E-Vec[Rcvr-Purpose-Ben] = Cur-App[[Default-Purpose-Ben];]
    170218 --> 170240
    170214 -- F --> 170240
    170222{If Result-Type = SUCCESS} -- T --> 170226[Evaluate all terminal node functions; Add up doer benefits to form Doer-Ben; Add up receiver benefits to form Rcvr-Ben; P-E-Vec[Doer-Purpose-Ben] = Doer-Ben; P-E-Vec[Rcvr-Purpose-Ben] = Rcvr-Ben;]
    170226 --> 170228{If Doer-Ben is beneficial or neutral}
    170228 -- T --> 170240
    170228 -- F --> 170230{If Rcvr-Ben is beneficial or neutral}
    170230 -- T --> 170240
    170230 -- F --> 170232[Inform the Communication Manager of implausible benefits for Cur-Purpose;]
    170232 --> 170224[170-Return = 170226; Call Cur-App[Purpose-Benefits-Classify-Fail, Ben-Purpose, Cur-Node, Result-Type, R-Add, 170-Return];]
    170222 -- F --> 170224
  
```

FIG. 22C

FIG. 22D

```

graph TD
    160010[Initialize Steps 12 and 18 of Natural Language Processor(s) 10; Initialize non-Textual Language Processors 40; NEX = 0; Error-Action = False; Delay-Out = Error-Out = False;] --> 160012{If Init-App is true}
    160012 -- F --> 160016[Send out greeting message(s) to output device(s);]
    160012 -- T --> 160014[Call Init-App-Process;]
    160014 --> 160018[Go to Step 12;]
    160016 --> 160018
    160018 --> 160020{If Error-Action is true and Cur-Clause ends a source input}
    160020 -- T --> 160022[Go to Error-Return;]
    160020 -- F --> 160030{If Delay-Out is true}
    160030 -- T --> 160032[NEX = NEX + 1; DELAY[PUR, NEX] = Err-Node; DELAY[E-S, NEX] = Error-Source; DELAY[CAN, NEX] = Cancel-Criteria; DELAY[Err-Out, NEX] = OUTLIST[Star-Pos to Next-Pos]; DELAY[Out-Criteria, NEX] = Output-Criteria; DELAY[ACTION, NEX] = Error-Act; Go to 160042;]
    160030 -- F --> 160034{If Error-Out is true}
    160034 -- T --> 160036[Error-Action = Error-Act;]
    160034 -- F --> 160042
    160036 --> 160038[Store a purpose entry at Context-Purpose-Set: Err-Node function, Err-Node, Error-Source, 1st error output clause pointer, ERROR-COM; Error-Out = False;]
    160038 --> 160040[Next-Out = OUTLIST[Star-Pos to Nex-Pos]; RETURN = Step 18; Call 200[Cur-Nat-Lang, Next-Out, Star-Pos, Nex-Pos, RETURN];]
    160040 --> 160054[Store a purpose entry at Context-Purpose-Set: Cur-Resp-Func, Cur-Resp-Pur-Address, Response-Object, 1st response clause, RESPONSE-SEL; Initialize Star-Pos; OUTLIST[Star-Pos to Nex-Pos] = Out-List[Init-Pos to Next-Pos];]
    160054 --> 160052{If there is a response}
    160052 -- T --> 160056[Go to Step 18;]
    160052 -- F --> 160044{If Cur-Clause ends a sentence}
    160044 -- T --> 160046[Evaluate the cancel criteria in DELAY[CAN, X]; Remove satisfied entries; Compress DELAY; Adjust NEX; Evaluate the delay criteria in DELAY[Out-Criteria, X] until a satisfied entry, S, is found; OUTLIST = DELAY[Err-Out, S]; Err-Node = DELAY[PUR, S]; Error-Source = DELAY[E-S, S]; E-A = DELAY[ACTION, S]; Remove the S entry;]
    160044 -- F --> 160052
    160046 --> 160048{If output criteria is satisfied}
    160048 -- T --> 160050[Error-Action = E-A;]
    160048 -- F --> 160054
    160050 --> 160038
    160000{If current Invocation-Opcode = INIT} -- T --> 160010
    160000 -- F --> 160002{If current Invocation-Opcode = CONTINUE}
    160002 -- T --> 160020
    160002 -- F --> 160004{If current Invocation-Opcode = "inform" word sense no.}
    160004 -- T --> 160008[Go to 160200;]
    160004 -- F --> 160006[Go to 160-Return;]

```

FIG. 23A

```

graph TD
    160200["Cur-Error = The 'inform' word sense number;  
Err-Node = Cur-Error's owned purpose realization entry in 130;  
Dynamic-Purpose-Name = Cur-Error's error function address;"] --> 160202{"If  
there is not an  
ERROR function  
at Err-Node"}
    160202 -- F --> 160240{"If  
Dynamic-Purpose-Name is a dynamic  
purpose address"}
    160202 -- T --> 160204["E-Output = Err-Node  
Err-Out function  
address;"]
    160240 -- F --> 160246["160-Return = 160244;  
Go to Dynamic-Purpose-Name[  
160-Return];"]
    160240 -- T --> 160242["160-Return = 160244;  
RS = False;  
Go to 140[DYNAMIC,  
Dynamic-Purpose-  
Name, RS, 160-Return];"]
    160204 --> 160206{"If  
Err-Node has  
an Err-Out  
function"}
    160206 -- F --> 160208["Inform the Communication  
Manager of a failure  
processing error;"]
    160206 -- T --> 160210["Replace variable references in  
E-Output clauses with values;  
Star-Pos = Next unused  
position in OUTLIST;  
Store E-Output clauses starting  
at OUTLIST[Star-Pos];"]
    160210 --> 160212{"If  
Err-Node has  
a DELAY-O  
function"}
    160212 -- T --> 160214["Delay-Out = True;  
Output-Criteria =  
DELAY-O address;  
Cancel-Criteria  
CANCEL criteria from  
Err-Node;  
Error-Out = False;"]
    160212 -- F --> 160216["Delay-Out = False;  
Error-Out = True;"]
    160214 --> 160218{"If  
Err-Node has  
an Err-Response-  
Act function"}
    160218 -- F --> 160228["Error-Act = False;"]
    160218 -- T --> 160220["Error-Act = True;  
Dynamic-Pur-Name =  
Err-Node's  
Err-Response-Act  
function;"]
    160220 --> 160222{"If  
Dynamic-Pur-  
Name is a dynamic  
purpose address"}
    160222 -- T --> 160224["160-Return = 160244;  
RS = Err-Node's  
Res-Value;  
Error-Return =  
140[DYNAMIC,  
Dynamic-Pur-Name,  
RS, 160-Return];"]
    160222 -- F --> 160226["160-Return = 160244;  
Error-Return =  
Dynamic-Pur-Name[  
160-Return];"]
    160224 --> 160230["Call 160[CONTINUE];"]
    160226 --> 160230
    160228 --> 160230
    160230 --> 160230
  
```

FIG. 23B

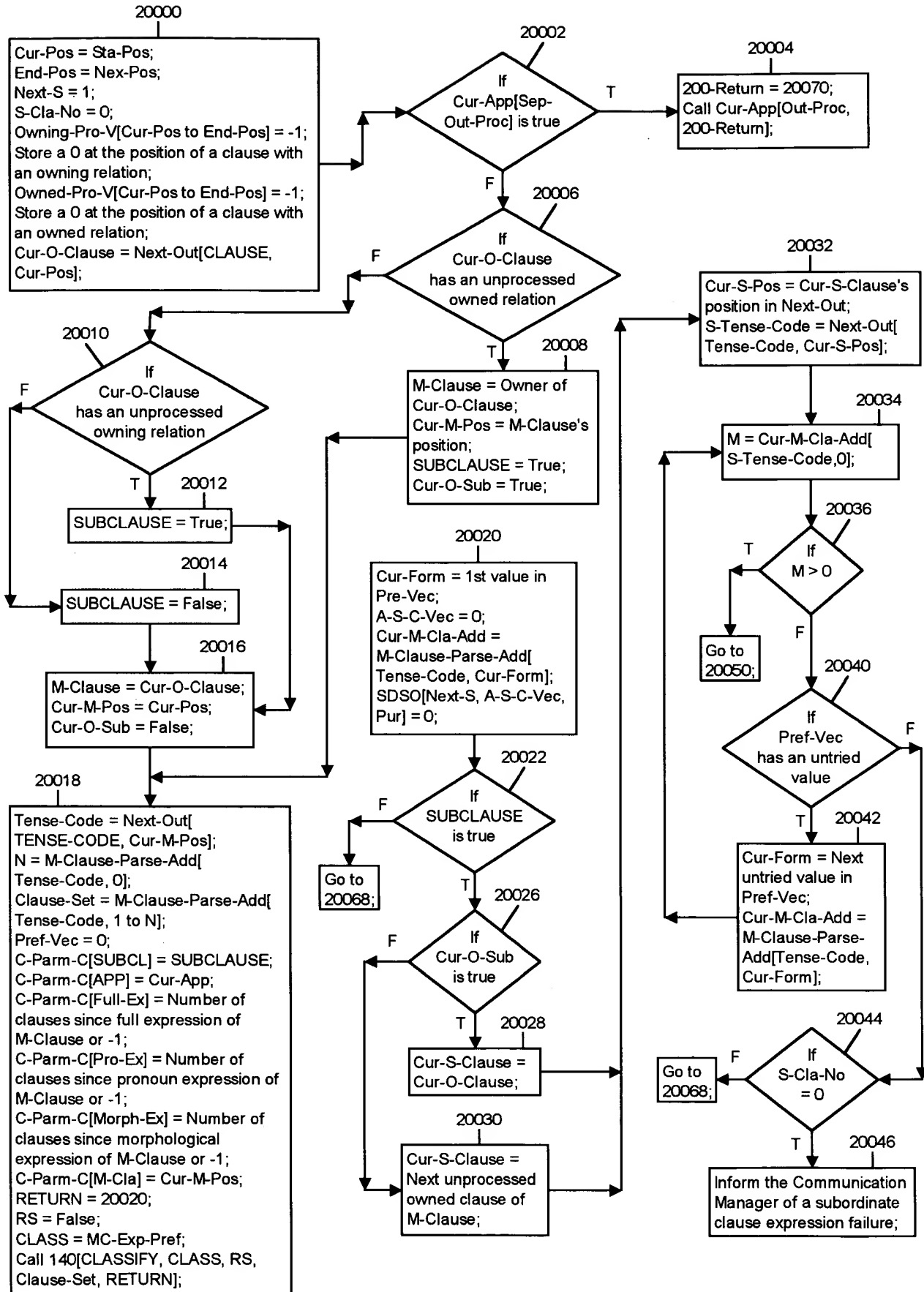


FIG. 24A

```

graph TD
    20050[20050  
S-Clause-Set = Cur-M-Cla-Add[  
Tense-Code, 1 to M];  
Pref-S-Form = 0;  
Pref-S-Pos = 0;  
C-Parm-S[A-Pos] = A-S-C-Vec;  
C-Parm-S[APP] = Cur-App;  
C-Parm-S[Purpose] =  
Next-Out[Cur-Rel, Cur-S-Pos];  
C-Parm-S[S-Clause-No] = S-Cla-No;  
C-Parm-S[Selected-Sub-Pur-Set] =  
SDSO[Next-S, A-S-C-Vec, Pur];  
C-Parm-S[Full-Ex] = Number of  
clauses since full expression of  
Cur-S-Clause or -1;  
C-Parm-[Pro-Ex] = Number of  
clauses since pronoun expression of  
Cur-S-Clause or -1;  
C-Parm-S[Morph-Ex] = Number of  
clauses since morphological  
expression of Cur-S-Clause or -1;  
C-Parm-S[S-Cla] = Cur-S-Pos;  
RETURN = 20052;  
RS = False;  
CLASS = SC-Exp-Pref;  
Call 140[CLASSIFY, CLASS, RS,  
Clause-Set, RETURN];] --> 20052{20052  
If  
Pref-S-Form  
= 0}
    20052 -- T --> 20056{20056  
If  
S-Cla-No  
= 0}
    20052 -- F --> 20054[20054  
Cur-S-Add = Cur-M-Cla-Add[S-Tense-Code,  
Pref-S-Form];  
S-Cla-Pos = Cur-M-Cla-Add[ S-Tense-Code,  
Pref-S-Pos];  
Pref-Imp-Vec = Cur-S-Add[IMP];  
Append Pref-Imp-Vec to  
SDSO[Next-S, S-Cla-Pos, Pref-Imp];  
Append Cur-S-Add to  
SDSO[Next-S, S-Cla-Pos, ADD];  
Append Cur-S-Pos to  
SDSO[Next-S, S-Cla-Pos, N-O-Pos];  
Owned-Pro-V[Cur-S-Pos] = 1;  
SDSO[Next-S, S-Cla-Pos, PUR] =  
C-Parm-S[Purpose];  
A-S-C-Vec[S-Cla-Pos] = 1;  
S-Cla-No = S-Cla-No + 1;]
    20056 -- T --> 20040[Go to  
20040;]
    20056 -- F --> 20060{20060  
If  
M-Clause has  
an untried and  
unprocessed  
owning relation}
    20060 -- T --> 20062[20062  
Cur-S-Pos = Next  
untried, unprocessed  
owned clause of  
M-Clause;  
Pref-Vec = All values  
as tried;  
Go to 20032;]
    20060 -- F --> 20064{20064  
If  
M-Clause has  
an unprocessed  
owning relation}
    20064 -- T --> 20068[20068  
SDSO[Next-S, M-Cla-Pos, ADD] =  
Cur-M-Cla-Add;  
SDSO[Next-S, M-Cla-Pos, N-O-Pos] =  
Cur-M-Pos;  
SDSO[Next-S, M-Cla-Pos, Pref-Imp] =  
Cur-M-Cla-Add[IMP];  
A-S-C-Vec[S-Cla-Pos] = 1;  
COORD = False;  
Go to 20070;]
    20064 -- F --> 20066[20066  
Owning-Pro-V[Cur-  
M-Pos] = 1;]
    20066 --> 20068

```

FIG. 24B

```

graph TD
    20070[SDSO-Pos = Next "1" position in A-S-C-Vec;  
IC = True;  
Cur-I-V = First address at  
SDSO[Next-S, SDSO-Pos, Pref-Imp];  
Cur-Cla-Add = First address at  
SDSO[Next-S, SDSO-Pos, ADD];  
Nex-O-Pos = First address at  
SDSO[Next-S, SDSO-Pos, N-O-Pos];  
Clear such addresses at SDSO;  
SDS[Current] = Next SDS entry;] --> 20072{If  
Cur-Cla-Add  
is a morphologically  
realized clause}
    20072 -- T --> 20074[Cur-Imp-V = Cur-I-V;  
Morph-Cla = True;  
Cur-S-R = Sentence  
role of Cur-Cla-Add;  
In-Call = False;]
    20072 -- F --> 20076[Cur-S-R-Add = Address of next unprocessed sentence role  
at Cur-Cla-Add in 30;  
Morph-Cla = In-Call = False;  
Cur-S-R-Head = Head of sentence role phrase at  
Cur-S-R-Add;  
Cur-S-R = The sentence role;  
Cur-Source = 1st entry at Next-Out[Cur-S-R, Nex-O-Pos];  
Cur-Imp-V = Cur-I-V[Cur-Source];  
Cur-Source-Head = Head, modifiers, and any function word  
vector of Cur-Source;]
    20074 --> 20078{If  
Cur-Source has  
a morphological  
implementation  
vector}
    20078 -- T --> 20079[Append the  
morphological  
vector to  
Cur-Imp-V;]
    20078 -- F --> 20081[Base-Word-Set = 0;  
M-BW-Set = 0;]
    20079 --> 20080[Spec-Morph-WV = True;]
    20080 --> 20086[Morph-Call = False;  
Fail-Return = False;  
Go to 200100;]
    20081 --> 20082{If  
Cur-S-R-Head  
type = Cur-Source-  
Head type}
    20082 -- F --> 20084[Spec-Morph-WV = False;]
    20082 -- T --> 20088{If  
Cur-Source  
is store in  
120}
    20084 --> 20086
    20088 -- T --> 20089[Ellip-Call = False;  
Coord-Check = False;  
Sentence-Check = False;  
Mod-Check = False;  
Go to 200200;]
    20088 -- F --> 20090{If  
Cur-S-R-Head  
type is a  
noun}
    20090 -- T --> 20091{If  
In-Call  
is true}
    20091 -- T --> 20093[U-Mod-C = Cur-App[U-M];  
Init-Head = Back-Track = True;  
Fail-Return = Fail-C = M-Word = Alt-Real = False;  
Entry-No = N-Mod = Sep-Mod = 0;  
MOD = 1;  
Cur-S-R[MODIFIER, Entry-No, MOD] = Cur-Source-Head's word  
sense number;  
Cur-S-R[RELATION, Entry-No, MOD] = Cur-Source's sentence role;  
Go to 200300;]
    20091 -- F --> 20090
    20090 -- F --> 20094{If  
Cur-S-R-Head  
type is a  
verb}
    20094 -- T --> 20095[In-Call = A-Call = False;  
Go to 200800;]
    20094 -- F --> 20096[In-Call = False;  
Go to 200940;]

```

FIG. 24C

FIG. 24D


```

graph TD
    200200[C-Parm-E/P[Object-V, 1 to 5] = False;] --> 200202{If Coord-Check is true and Sentence-Check is true}
    200202 -- T --> 200204[C-Parm-E/P[Object-V, Coord-Sentence] = True; E/P-Obj = SDS[Current];]
    200202 -- F --> 200206{If Coord-Check is true}
    200206 -- T --> 200208[C-Parm-E/P[Object-V, Coord-Phrase] = True; E/P-Obj = Cur-S-R;]
    200206 -- F --> 200210{If Sentence-Check is true}
    200210 -- T --> 200212[C-Parm-E/P[Object-V, Sentence] = True; E/P-Obj = SDS[Current];]
    200210 -- F --> 200214{If Mod-Check is true}
    200214 -- T --> 200216[C-Parm-E/P[Object-V, Modifier] = True; E/P-Obj = Check-Mod;]
    200214 -- F --> 200218[C-Parm-E/P[Object-V, Phrase] = True; Cur-S-R[TEXT, text range] = Text from 120; E/P-Obj = Cur-S-R[TEXT, text range];]
    200216 --> 200220[C-Parm-E/P[Full-Ex] = Number of phrases since full expression of E/P-Obj or -1; C-Parm-E/P[Pro-Ex] = Number of phrases since pronoun expression of E/P-Obj or -1;]
    200218 --> 200220
    200204 --> 200222[C-Parm-E/P[APP] = Cur-App; Ellip-Out-Pos-V = 0; Ellip-Trans-Pos-M = 0; Pro-Ex-Pos-V = 0; RETURN = 200230; RS = False; CLASS = ELLIP/PRO-Ex-Suitability; Call 140[CLASSIFY, CLASS, RS, E/P-Obj, RETURN];]
    200208 --> 200222
    200212 --> 200222
    200216 --> 200222
    200220 --> 200222
    200222 --> 200230{If Ellip-Out-Pos-V = 0}
    200230 -- T --> 200234{If Ellip-Trans-Pos-M = 0}
    200230 -- F --> 200232[Remove the text from E/P-Obj for each position in Ellip-Out-Pos-V;]
    200232 --> 200234
    200234 -- T --> 200240{If Pro-Ex-Pos-V = 0}
    200234 -- F --> 200236[For each non-zero pair in Ellip-Trans-Pos-M: transfer text from the source component to the E/P-Obj position component;]
    200236 --> 200240
    200240 -- T --> 200250{If Ellip-Call is true}
    200240 -- F --> 200242[For each sentence role in Pro-Ex-Pos-V: Pro-Ex = Pro-Select[sentence role]; Store each Pro-Ex at its position in E/O-Obj;]
    200242 --> 200250
    200250 -- T --> 200252[Return to caller;]
    200250 -- F --> 200704[Go to 200704;]
  
```

FIG. 24E

```

graph TD
    200300{200300  
If  
Cur-Source has  
a text realization} -- T --> 200302[200302  
D-MAX = Number of modifiers of the most  
recent reference of Cur-Source;  
Cur-S-R[TEXT, -D-MAX to 0] =  
Cur-Source's most recent reference;  
Go to 200700;]
    200300 -- F --> 200304{200304  
If  
Cur-App[  
Noun-Ex] is  
true}
    200304 -- T --> 200306[200306  
200-Return = 200700;  
Call Cur-App[Noun-Ex-Proc,  
Cur-Source-Head,  
200-Return];]
    200304 -- F --> 200308{200308  
If  
Cur-Source is  
unique in 120 for its  
word sense number  
and its assigned  
modifiers}
    200308 -- T --> 200314{200314  
If  
Cur-Source-Head  
has type indicating  
modifiers and does not  
have an assigned  
type modifier}
    200308 -- F --> 200310{200310  
If  
Cur-Source-Head  
is a general reference  
or if U-Mod-C is  
false}
    200314 -- T --> 200316[200316  
N-Mod = N-Mod + 1;  
Cur-S-R[MODIFIER/RELATION,  
Entry-No + 1, N-Mod] = Type  
indicating word sense number/  
type indicating relation address;]
    200314 -- F --> 200318{200318  
If  
Cur-Source-  
Head has assigned  
modifiers}
    200316 --> 200318
    200318 -- T --> 200320[200320  
A-Mod = Number of direct assigned modifiers;  
Store modifier word sense numbers/relation  
address at Cur-S-R[MODIFIER/RELATION,  
Entry-No + 1, N-Mod + 1 to N-Mod + A-Mod];  
N-Mod = N-Mod + A-Mod;]
    200318 -- F --> 200322{200322  
If  
N-Mod  
= 0}
    200320 --> 200322
    200322 -- T --> 200324[200324  
D-Mod = True;]
    200322 -- F --> 200326[200326  
D-Mod = False;]
    200324 --> 200312
    200326 --> 200312
    200310 -- T --> 200316
    200310 -- F --> 200308
    200312[200312  
Select an untried uniqueness setting modifier  
phrase of Cur-Source-Head which sets a  
unique state value or a unique relation to  
word sense numbers in 120 with the same  
identification number and type number;  
Modifiers are selected according to  
Cur-App[Mod-Ord-Policy];  
N-Mod = N-Mod + 1;  
Store modifier word sense numbers/relation  
address at Cur-S-R[MODIFIER/RELATION,  
Entry-No + 1, N-Mod];] --> 200310
  
```

FIG. 24F

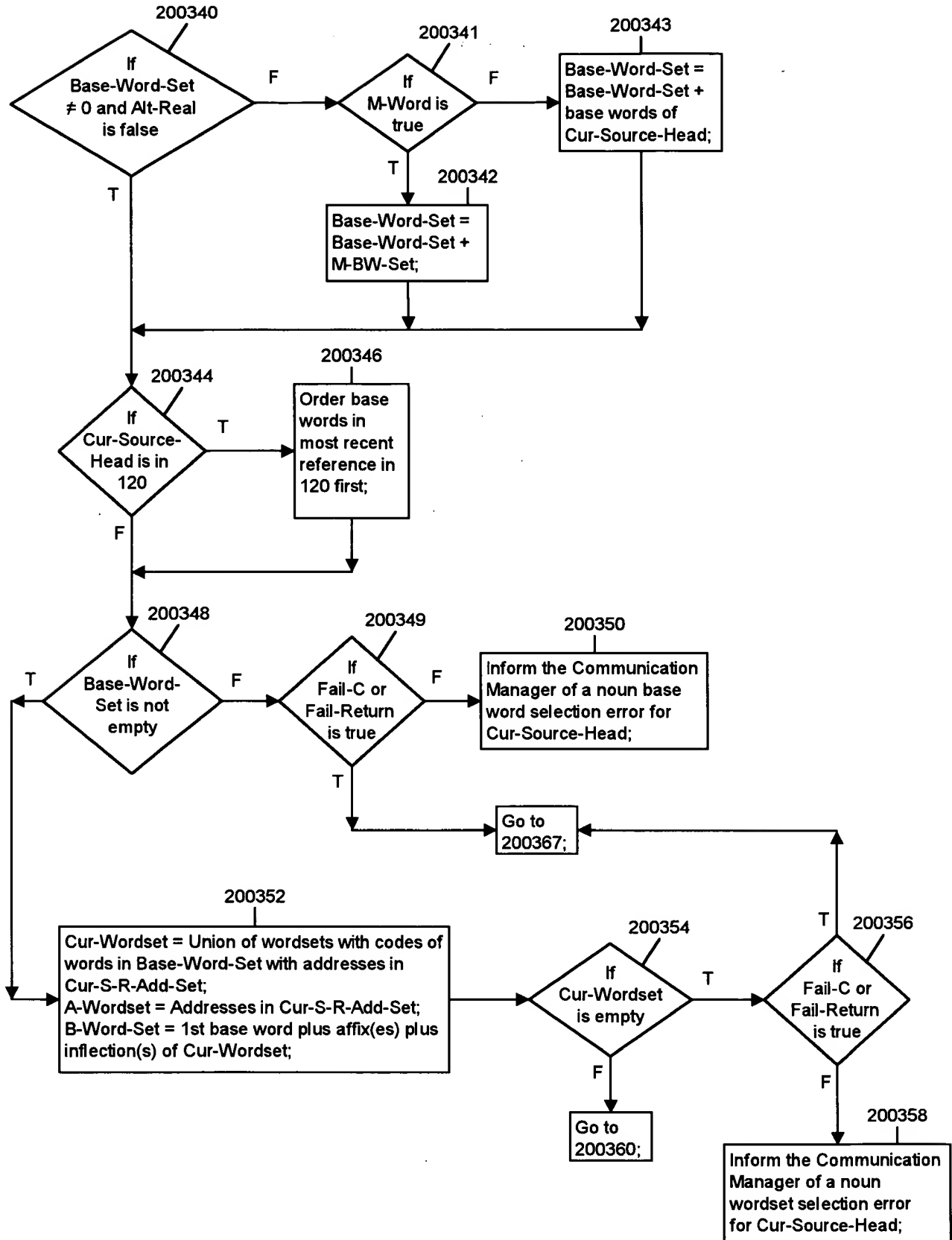


FIG. 24G

```

graph TD
    200362[200362  
Cur-F-Word-V = Function word application vector at  
Cur-Source-Head;  
For each wordset in Cur-Wordset: access the wordset's  
function word application vector at its Cur-S-R-Add-Set  
address;  
AND the wordset's vector with Cur-F-Word-V;  
remove wordsets which have a result vector not equal  
to Cur-F-Word-V;  
Cur-S-R[Fun-W, Entry-No, MOD] = Cur-F-Word-V;]
    200360{200360  
If  
Cur-Source-Head  
has a function word  
modifier}
    200364{200364  
If  
Cur-Wordset  
is empty}
    200365{200365  
If  
Fail-C or  
Fail-Return  
is true}
    200366[200366  
Inform the  
Communication  
Manager of a function  
wordset realization error  
for Cur-Source-Head;]
    200368{200368  
If  
Cur-Source-Head  
has a clause  
modifier}
    200370[200370  
Remove all wordsets and their  
associated entries which do not  
allow the types of assigned  
clause modifiers form  
Cur-Wordset, A-Wordset, and  
B-Word-Set;]
    200372{200372  
If  
Cur-Wordset  
is empty}
    200373{200373  
If  
Fail-C or  
Fail-Return  
is true}
    200374[200374  
Inform the  
Communication  
Manager of a clause  
modifier realization error  
for Cur-Source-Head;]
    200376[200376  
N-WS = Number of wordsets;  
Cur-S-R[WORDSET/ADDRESS/  
BWORD, Entry-No, MOD,  
1 to N-WS] = Cur-Wordset/  
A-Wordset/ B-Word-Set;  
Cur-S-R[WORDSET, Entry-No,  
MOD, 0] = N-WS;  
Base-Word-Set = 0;]
    200377{200377  
If  
Init-Head  
is true}
    200378[200378  
Init-Head =  
False;  
Go to 200380;]
    200369[200369  
FAILED =  
True;  
Go to 200430;]
    200367{200367  
If  
Fail-C is  
true}
    200371[200371  
FAIL = True;  
Go to  
200-Return;]
    200430[200430  
Go to  
200430;]

    200362 --> 200360
    200360 -- F --> 200364
    200360 -- T --> 200362
    200364 -- T --> 200365
    200364 -- F --> 200368
    200365 -- F --> 200366
    200365 -- T --> 200373
    200366 --> 200373
    200368 -- T --> 200370
    200368 -- F --> 200376
    200370 --> 200372
    200372 -- T --> 200373
    200372 -- F --> 200376
    200373 -- F --> 200374
    200373 -- T --> 200369
    200374 --> 200369
    200376 --> 200377
    200377 -- T --> 200378
    200377 -- F --> 200430
    200378 --> 200369
    200369 --> 200367
    200367 -- F --> 200371
    200367 -- T --> 200369

```

FIG. 24H

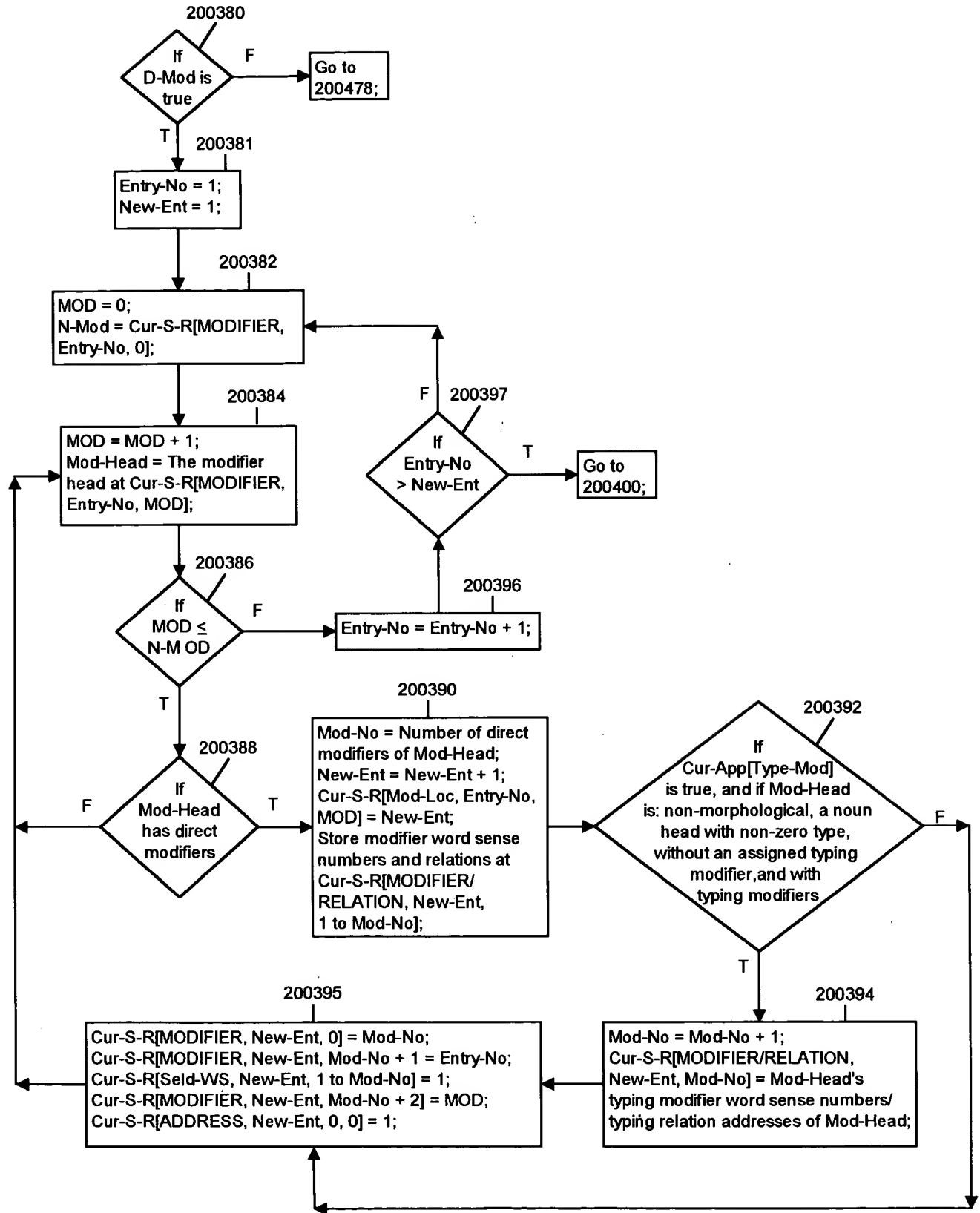


FIG. 24 I

```

graph TD
    200400[Entry-No = 1;  
MOD = 1;  
Head-WS = 1;] --> 200402[N-Mod = Cur-S-R[MODIFIER, Entry-No, 0];  
M-Ent = Cur-S-R[MODIFIER, Entry-No, N-Mod + 1];  
M-Mod = Cur-S-R[MODIFIER, Entry-No, N-Mod + 2];  
Modife-WS-No = Cur-S-R[WORDSET, M-Ent, M-Mod, 0];  
Cur-WS = Cur-S-R[WORDSET, M-Ent, M-Mod, Head-WS];  
INC = Number of Cur-WS addresses;]
    200402 --> 200404[M-Word = Alt-Real = False;  
M-BW-Set = 0;  
Cur-Source-Head = Cur-S-R[MODIFIER, Entry-No, MOD];  
Spec-Vec = Cur-Source-Head's implementation vector or a standard vector if none;  
Cur-Rel = Cur-S-R[MODIFIER, Entry-No, MOD];]
    200404 --> 200406{If  
Cur-Source-Head  
needs morphological  
processing}
    200406 -- T --> 200471[Go to  
200471;]
    200406 -- F --> 200407{If  
Cur-Rel has a  
morphological  
realization}
    200407 -- T --> 200408[Alt-Real = Fail-Return = Morph-Call = True;  
SOURCE = Word sense number type of Cur-Source-Head;  
Func-Type = Function type of Cur-Rel morphological  
realizations;  
Morph-W-S = Cur-Source-Head's word sense number;  
Spec-Vec = Spec[M-Rel] or a standard vector if none;  
Spec-Morph-W = Spec-Vec[REL];  
RETURN = 200410;  
Call 200[Spec-Vec[ADD], Spec-Vec, RETURN];]
    200408 --> 200414[Cur-S-R-Add-Set = NULL;]
    200407 -- F --> 200414
    200414 --> 200416{If  
Cur-Rel has a  
non-prepositional realization  
and if Cur-Source-Head does  
not have a noun function  
word vector}
    200416 -- T --> 200418[Cur-S-R-Add-Set = The non-prepositional  
Cur-Rel and Cur-Source-Head compatible,  
unmarked, addresses of Cur-WS at  
Cur-S-R[ADDRESS, M-Ent, M-Mod,  
Head-WS to Head-WS + INC - 1];]
    200416 -- F --> 200420{If  
Cur-Rel has a  
prepositional  
realization}
    200420 -- T --> 200422[Cur-S-R-Add-Set = Cur-S-R-Add-Set plus  
the prepositional Cur-Rel and  
Cur-Source-Head compatible, unmarked,  
addresses of Cur-WS at  
Cur-S-R[ADDRESS, M-Ent, M-Mod,  
Head-WS to Head-WS + INC - 1];]
    200420 -- F --> 200424{If  
Cur-S-R-Add-Set  
is NULL}
    200424 -- T --> 200448[Go to  
200448;]
    200424 -- F --> 200425{If  
Cur-Rel is an  
adjective modification  
relation}
    200425 -- T --> 200427[Function word application vector =  
DEG-ADV[Cur-Source-Head,  
Cur-Source-Head's state value];]
    200425 -- F --> 200428[Fail-C = True;  
FAILED = False;  
Go to 200340;]
    200427 --> 200428
    200418 --> 200422
    200422 --> 200418
    200418 --> 200410{If  
FAIL is false or  
M-BW-Set = 0 or  
M-Word is  
false}
    200410 -- T --> 200414
    200410 -- F --> 200448

```

FIG. 24J

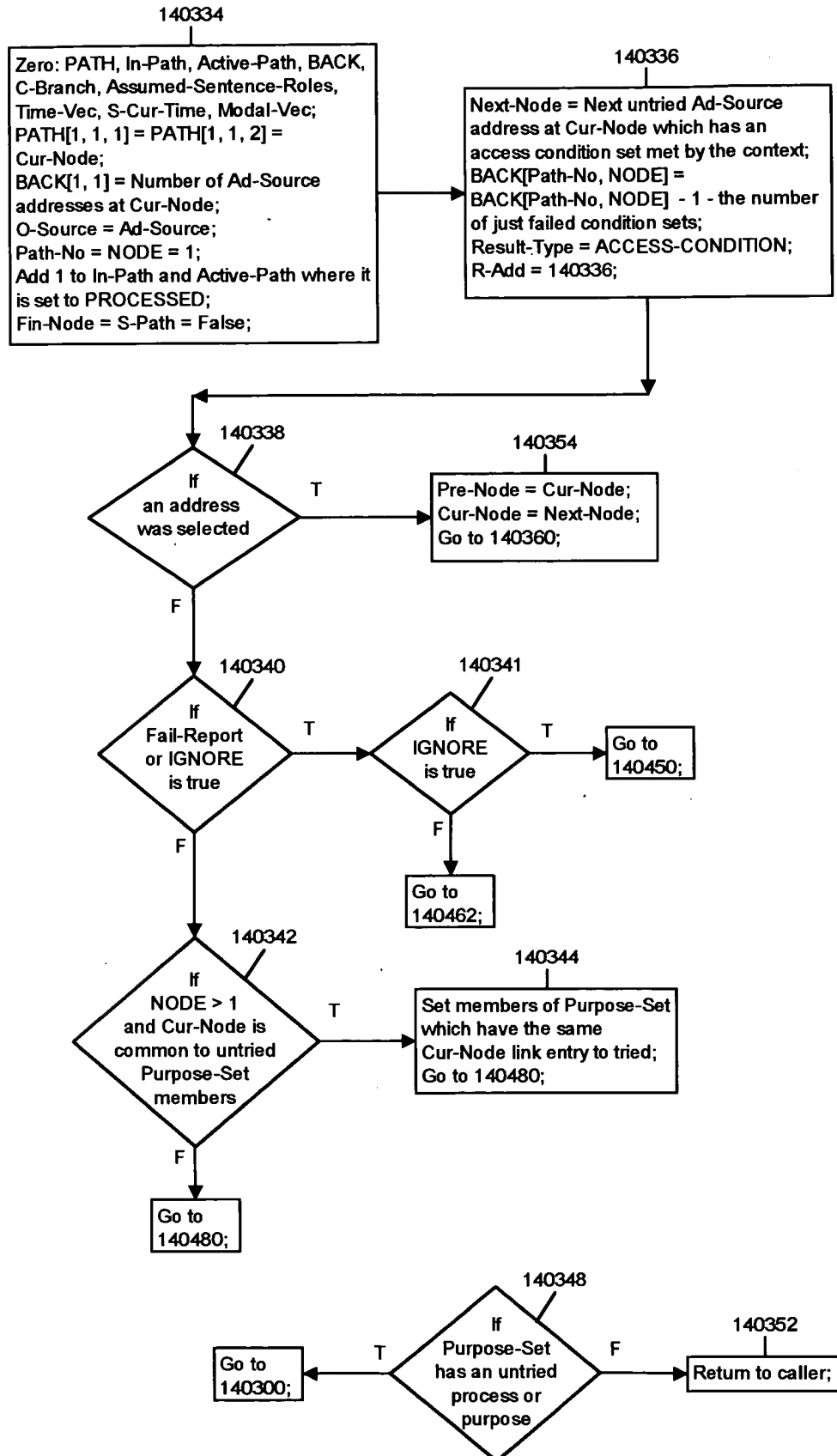


FIG. 21K

[illegible]

FIG. 24L


```

graph TD
    200520{If P-Pos > 0} -- F --> 200674[Go to 200674]
    200520 -- T --> 200524[M-Lim = 1;  
Prep-Real = T-Prep-Real = Cur-Prep-Set = 0;  
Dn-Mod = Cur-S-R[MODIFIER, 1, 0];]
    200524 --> 200526[Cur-Mod = M-Order[1, M-Lim];  
L-Ent = Cur-S-R[Mod-Loc, 1, Cur-Mod];  
Nx-Mod = Cur-Mod;]
    200530[M-Lim = M-Lim + 1;] --> 200526
    200526 --> 200528{If L-Ent = 0}
    200528 -- T --> 200529[Up-Ent[Cur-Mod] = 1;]
    200528 -- F --> 200532{If Cur-Mod = Dn-Mod}
    200532 -- T --> 200540[U-Ent = New-Ent;]
    200532 -- F --> 200534[Nx-Mod = Nx-Mod + 1;  
U-Ent = Cur-S-R[Mod-Loc, 1, Nx-Mod] - 1;]
    200534 --> 200536{If U-Ent < 0}
    200536 -- T --> 200538{If Nx-Mod = Dn-Mod}
    200538 -- T --> 200540
    200538 -- F --> 200534
    200536 -- F --> 200542[Up-Ent[Cur-Mod] = U-Ent;  
Cur-Prep-Set = Each P-Pos with  
L-Ent < PREP[P-Pos, E] < U-Ent;  
T-P-S = Cur-Prep-Set;]
    200542 --> 200544{If PREP has Cur-Mod stored  
for a P-Pos}
    200544 -- T --> 200546[PREP[P-Pos, E/M] = 0 for  
Cur-Mod's P-Pos;  
Cur-Prep-Set = Cur-Mod's P-Pos;  
T-Prep-Real[1, Cur-Mod] = 1;]
    200544 -- F --> 200548[Go to 200548]
    200546 --> 200548
    200529 --> 200544
    200540 --> 200544

```

FIG. 24M

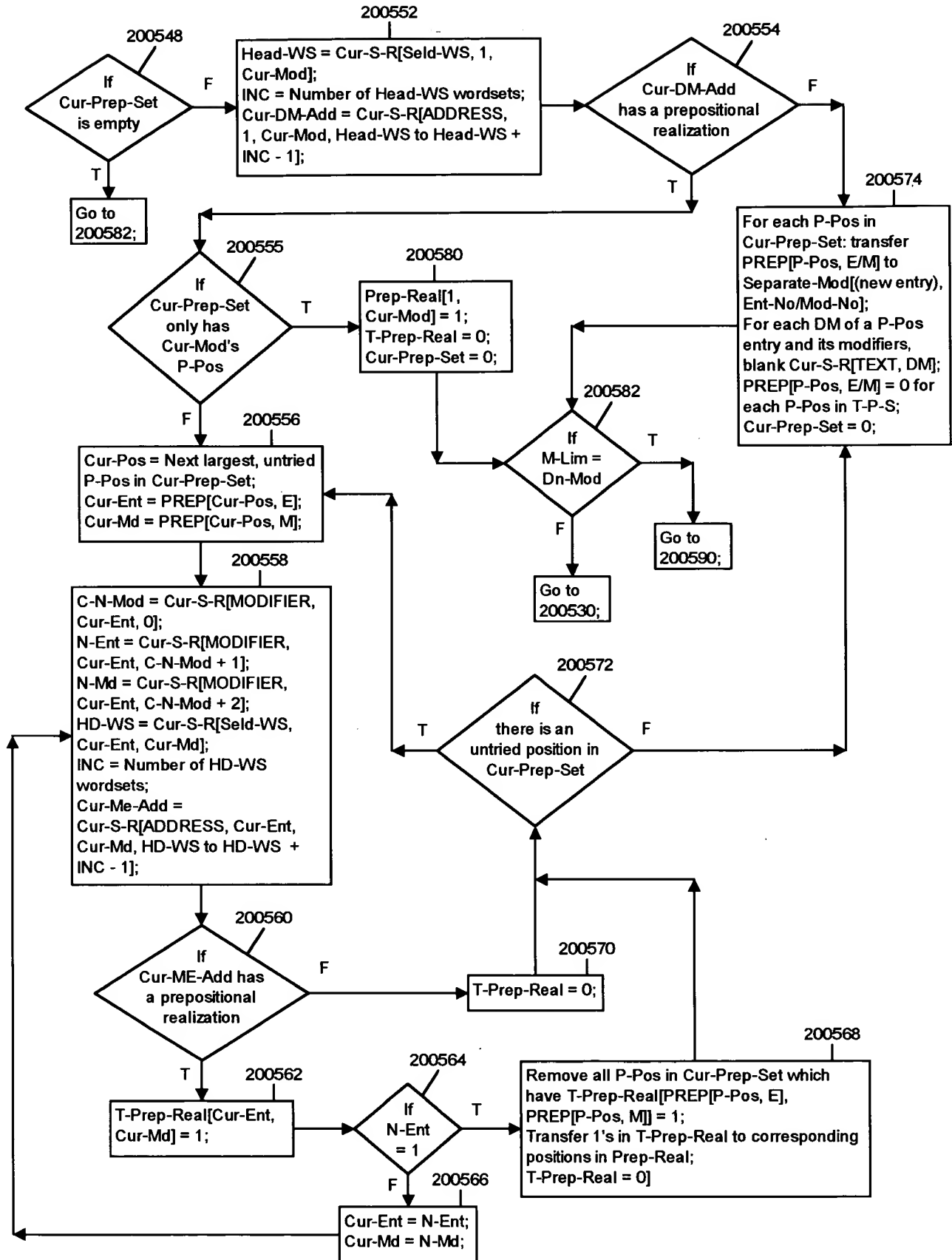


FIG. 24N

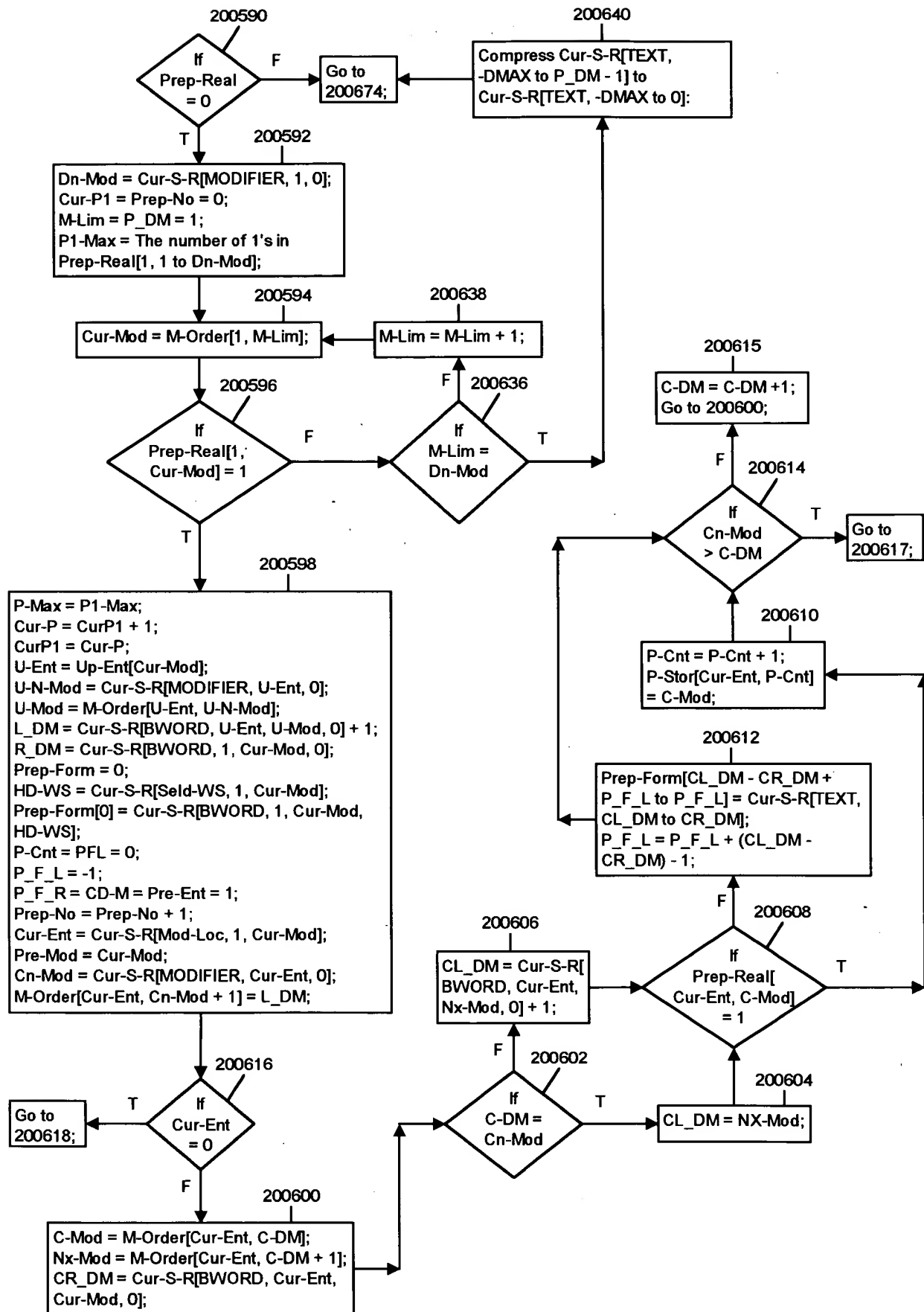


FIG. 240

```

graph TD
    200617{If Pre-Ent = 1} -- T --> 200619[PFL = MIN[PFL, P_F_L + 1];]
    200617 -- F --> 200618
    200619 --> 200618
    200618[Cur-WS = Cur-S-R[WORDSET, Pre-Ent, Pre-Mod, HD-WS];  
B-Word-No = Cur-S-R[BWORD, Pre-Ent, Pre-Mod, HD-WS];  
INC = Number of Cur-WS's wordsets;  
Cur-Add = Prepositional partition in Cur-S-R[ADDRESS, Pre-Ent, Pre-Mod, HD-WS to HD-WS + INC - 1];  
C-WDSN = Cur-S-R[MODIFIER, Pre-Ent, Pre-Mod];  
Cur-DM = Cur-S-R[BWORD, Pre-Ent, Pre-Mod, 0];  
Det-Sel = False;  
Prep-Func = Function word of preposition selected with Cur-App[Prep-Sel] at Cur-Add;  
Prep-Text = Text associated with Prep-Func;] --> 200620
    200620{If Cur-S-R[TEXT, Cur-DM] has a function word} -- T --> 200622[Prep-Text = Prep-Text + function words at Cur-S-R[TEXT, Cur-DM];]
    200620 -- F --> 200624
    200622 --> 200626
    200624{If there is a determiner anomaly at B-Word-No for Cur-WS and Prep-Func} -- T --> 200626
    200624 -- F --> 200628
    200626[Cur-Det = The result of evaluating the anomaly function at B-Word-No;  
Det-Sel = True;] --> 200630
    200628[Cur-Rel = Cur-S-R[RELATION, Pre-Ent, Pre-Mod];  
Cur-FWS = The result of evaluating functions of Cur-Rel and Prep-Func for Det-Sel and C-WDSN to select a determiner and degree adverb;] --> 200630
    200630[Prep-Text = Prep-Text + Cur-FWS;] --> 200631
    200631{If P-Max > 1 and Cur-P = P-Max} -- T --> 200632[Prep-Text = "and" + Prep-Text;]
    200631 -- F --> 200633
    200632 --> 200635
    200633{If P-Max > 2} -- T --> 200634[Insert "," at Prep-Form[P_F_R];]
    200633 -- F --> 200635
    200634 --> 200635
    200635[Prep-Form[P_F_L + 1] = Prep-Text;  
Go to 200644;]

```

FIG. 24P

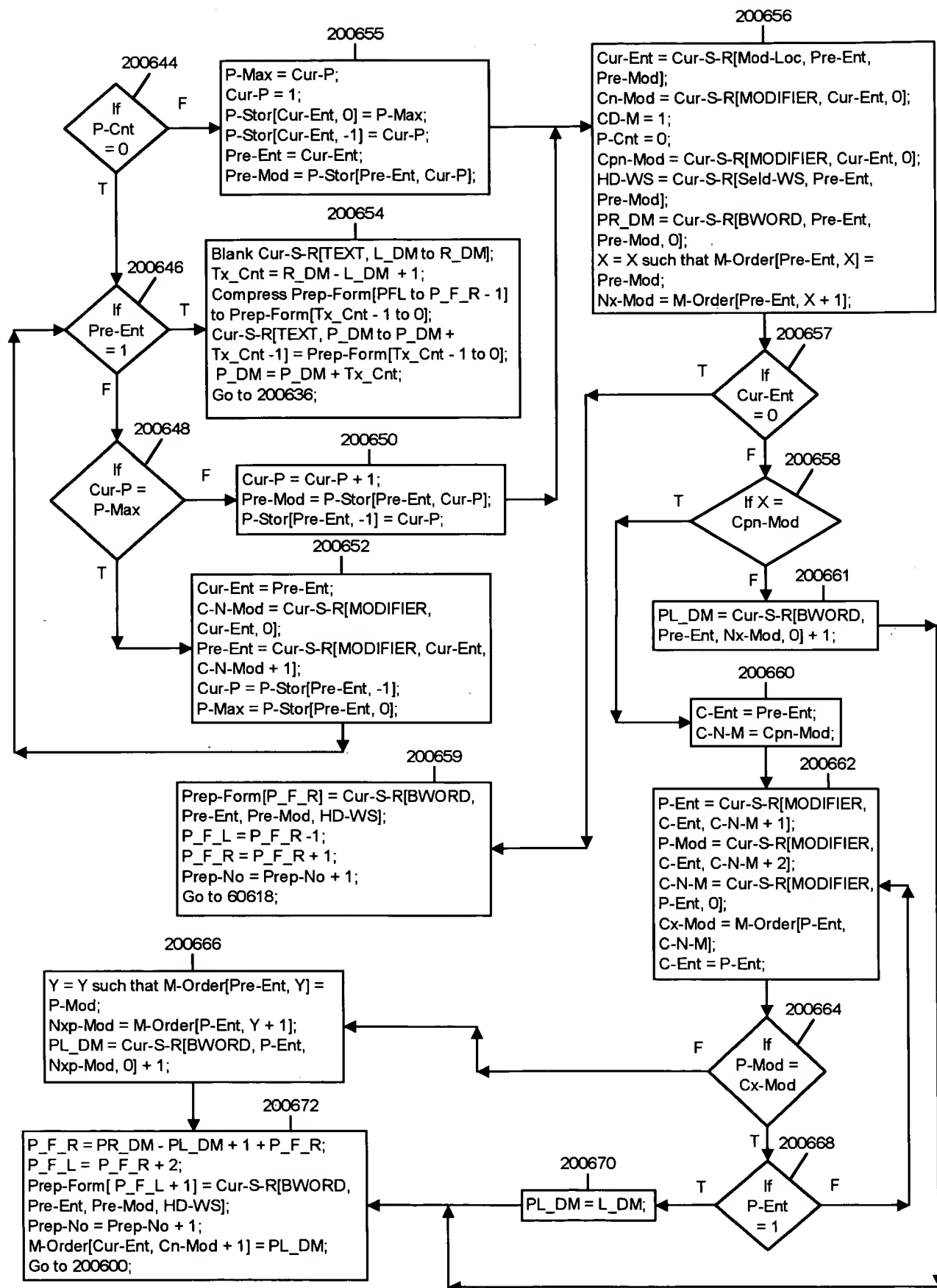


FIG. 24Q

```

graph TD
    200674{If Sep-Mod = 0}
    200676[Cur-Sep = 1;  
C-Parm-N[CLAU] = Cur-Cla-No;  
C-Parm-N[APP] = Cur-App;  
C-Parm-N[PREPS] = Prep-No;  
C-Parm-N[SEP-M] = Sep-Mod;  
C-Parm-N[S-R] = Sentence role of Cur-S-R;  
C-Parm-N[Cla-Mod] = Number of clauses modifying T-Cur-Source-Head;  
C-Parm-N[N-O-Pos] = Nex-O-Pos;  
RS = False;  
RETURN = 200682;  
CLASS = SEP-N-MOD-EXP-PREF;]
    200680[200680  
C-Parm-N[S-Mod-N] = Cur-Sep;  
Clause-Add = Sent-Role = TEMPLATE = 0;  
Sep-Mod-Ent = Separate-Mod[Cur-Sep, ENT-NO];  
Sep-Mod-Mod = Separate-Mod[Cur-Sep, MOD-NO];  
Sep-Mod-NMod = Cur-S-R[MODIFIER, Sep-Mod-Ent, 0];  
Sep-Mod-W = Cur-S-R[MODIFIER, Sep-Mod-Ent, Sep-Mod-Mod];  
Sep-ME-Ent = Cur-S-R[MODIFIER, Sep-Mod-Ent, Sep-Mod-NMod + 1];  
Sep-ME-Mod = Cur-S-R[MODIFIER, Sep-Mod-Ent, Sep-Mod-NMod + 2];  
Sep-ME-W = Cur-S-R[MODIFIER, Sep-ME-Ent, Sep-ME-Mod];  
Call 140[CLASSIFY, CLASS, RS, Sep-ME-W, RETURN];]
    200682[200682  
Cur-Sep = Cur-Sep + 1;  
Cur-Sep-Head = Sep-ME-W + (Sep-Mod-W + word sense numbers of all its modifiers);]
    200684[200684  
Next-Out[TEMPLATE[Sent-Role], TEMPLATE[N-O-P]] = Cur-Sep-Head;]
    200686{200686  
If Clause-Add = 0}
    200688[200688  
End-Pos = End-Pos + 1;  
Next-Out[Associated value names, End-Pos] = TEMPLATE[Stored Next-Out parameters and values];  
Cla-Pos = TEMPLATE[Clause-Realization-Position];  
Append TEMPLATE[Clause-Parse-Add] to SDSO[Next-S, Cla-Pos, ADD];  
Append TEMPLATE[Imp-Vec] to SDSO[Next-S, Cla-Pos, Pref-Imp];  
Append End-Pos to SDSO[Next-S, Cla-Pos, N-O-Pos];  
A-S-C-Vec[Cla-Pos] = 1;]
    200690{200690  
If Cur-Sep = Sep-Mod}
    200692{200692  
If In-Call is true}
    200694[200694  
Cur-S-R[TEXT-Len] = DMAX + 1;  
Return to caller;]
    200700[Go to 200700;]

    200674 -- T --> 200682
    200674 -- F --> 200676
    200676 --> 200680
    200680 --> 200686
    200682 --> 200684
    200684 --> 200686
    200686 -- T --> 200690
    200686 -- F --> 200688
    200688 --> 200690
    200690 -- F --> 200692
    200690 -- T --> 200694
    200692 -- F --> 200700
    200692 -- T --> 200694
    200694 --> 200680
    200700 --> 200674
  
```

Go to
200700;

```

graph TD
    200800{200800  
If  
Cur-Source has  
a text realization} -- T --> 200802[200802  
DMAX = Number of words in Cur-Source - 1;  
Cur-S-R[TEXT, -DMAX to 0] = Text at Cur-Source;  
Go to 200700;]
    200800 -- F --> 200804[200804  
V-Source-Head =  
Cur-Source-Head;]
    200804 --> 200806{200806  
If  
V-Source-Head  
has a Base-  
Word-Set}
    200806 -- F --> 200808[200808  
Base-Word-Set = Set of  
text base words and codes  
of V-Source-Head;]
    200806 -- T --> 200812{200812  
If  
V-Source-Head is in  
120}
    200808 --> 200810{200810  
If  
Base-Word-Set  
is empty}
    200810 -- T --> 200816[200816  
Inform the Communication Manager  
of a base word selection error for  
V-Source-Head;]
    200810 -- F --> 200812
    200812 -- T --> 200814[200814  
Order Base-Word-Set  
by the most recent first  
policy;]
    200812 -- F --> 200820[200820  
V-Base-Word-Set =  
Base-Word-Set;  
V-Wordset = 0;  
V-S-R-Add = Cur-S-R-Add;  
V-S-R = Cur-S-R;]
    200814 --> 200822[200822  
Direct adverbial modifiers of V-Source-Head  
are set to unprocessed;  
V-Wordset = Next, unprocessed wordset +  
codes of a word in V-Base-Wordset which is  
stored at an untried address at V-S-R-Add;]
    200822 --> 200824{200824  
If  
V-Wordset  
= 0}
    200824 -- T --> 200826[200826  
Inform the Communication Manager  
of a wordset selection  
error for V-Source-Head;]
    200824 -- F --> 200828{200828  
If  
V-Source-Head  
has modifiers}
    200828 -- T --> 200832[200832  
V-Mod-Add-Set = The set of addresses at  
V-S-R-Add[V-Wordset] of adverbial  
modifiers;  
V-Mod-Pos-Set[POS, POS-Range] = V  
where V = 1 for an allowed position, and  
V = 0 otherwise;  
V-Mod-Pos-Set[SUB, POS-Range, 0] = 0;]
    200828 -- F --> 200918[200918  
Go to]
    200832 --> 200834{200834  
If  
Cur-App[V-Ex]  
= True}
    200834 -- F --> 200838[200838  
Go to]
    200834 -- T --> 200836[200836  
200-Return = 200700;  
Call Cur-App[V-Ex-Pro,  
V-Source-Head, 200-Return];]
    200836 --> 200918

```

FIG. 24S

```

graph TD
    200838[200838  
V-Mod = The next unprocessed direct  
modifier adverbial subclass of  
V-Source-Head;  
V-Mod-Real-Add[0] =  
ADV-Sub-Real-Add-Func[V-Mod,  
Cur-App, ADD];  
V-Mod-Real-Len =  
ADV-Sub-Real-Add-Func[V-Mod,  
Cur-App, LEN];  
ADV-Add = -1;  
LY-WS = False;] --> 200840[200840  
ADV-Add = ADV-Add + 1;]
    200840 --> 200842{200842  
If  
ADV-Add >  
V-Mod-Real-  
Len}
    200842 -- T --> 200844{200844  
If  
LY-WS  
is true}
    200842 -- F --> 200846[200846  
Cur-Real = V-Mod-Real-  
Add[ADV-Add];]
    200844 -- F --> 200847[200847  
V-Wordset = 0;  
Go to 200822;]
    200844 -- T --> 200898[200898  
POS = POS-LY;  
Text-Form =  
LY-Text;]
    200846 --> 200848{200848  
If  
Cur-Real  
is a function  
word}
    200848 -- F --> 200860[200860  
Go to  
200860;]
    200848 -- T --> 200849{200849  
If  
wordset at Cur-Rel  
matches an entry of  
V-Mod-Add-Set}
    200849 -- F --> 200840
    200849 -- T --> 200850[200850  
Cur-V-WS = Wordset  
at Cur-Real;  
Text-Form = Text of  
Cur-V-WS;]
    200850 --> 200851[200851  
POS = Position number of  
1st V-Mod-Add-Set  
address  
which contains Cur-V-WS;]
    200851 --> 200852[200852  
AV-No = V-Mod-Pos-Set[SUB,  
POS, 0] + 1;  
V-Mod-Pos-Set[SUB, POS, 0] =  
AV-No;  
V-Mod-Pos-Set[SUB, POS,  
AV-No] = V-Mod;  
Tex-Len = The number of words  
in Text-Form;  
VAL = V-Mod's subclass value;]
    200852 --> 200854{200854  
If  
VAL is not  
typical for  
V-Mod}
    200854 -- F --> 200856[200856  
Deg-AV-Text = D-AV-Func[V-Mod,  
VAL, Cur-V-WS];  
Text-Form = Deg-AV-Text +  
Text-Form;  
Tex-Len = Tex-Len + Length of  
Deg-AV-Text;]
    200854 -- T --> 200858[200858  
V-Mod-Pos-Set[TEXT, POS, AV-No,  
-Tex-Len + 1 to 0] = Text-Form;  
V-Mod-Pos-Set[TEXT, POS, AV-No,  
0] = Text-Len;]
    200858 --> 200859{200859  
If  
there is an  
unprocessed  
modifier}
    200859 -- T --> 200838
    200859 -- F --> 200918[200918  
Go to  
200918;]

```

FIG. 24T


```

graph TD
    200860{200860  
If  
Cur-Real is a  
morphological  
realization}
    200870[200870  
Mod-Type =  
ADVERBIAL;  
Go to 200900;]
    200862[200862  
V-Vec = Cur-Real's IMP vector  
or a standard vector if none;  
Spec-Morph-W = V-Vec[STAT];  
Cur-Source-Head = V-Mod's  
source word sense number;  
Cur-S-R-Head type =  
ADVERB;  
Fail-Return = True;  
Cur-S-R =  
ADVERBIAL-MODIFIER;  
Morph-Call = True;  
Entry-No = 1;  
200-Return = 200864;  
Call 200[200100, V-Vec,  
200-Return];]
    200864[200864  
Fail-Return =  
False;]
    200866{200866  
If  
FAIL is  
false}
    200840_1[200840;]
    200868[200868  
AV-Wordset = Union of  
wordsets + affixes of Base-  
Word-Set entries which have  
an entry in V-Mod-Add-Set;  
BW-SET = Base words +  
affixes associated with a  
wordset in AV-Wordset;]
    200872[200872  
Cur-V-WS = Next untried  
wordset in AV-Wordset;  
POS-Vec = 1's at vector  
positions corresponding to  
modifier position numbers  
whose address at  
V-Mod-Add-Set has  
Cur-V-WS;]
    200874{200874  
If  
POS-Vec  
is zero}
    200876{200876  
If  
AV-Wordset  
has an untried  
wordset}
    200840_2[200840;]
    200878[200878  
POS = Next  
untried position  
in POS-Vec;]
    200879{200879  
If  
A-Call is  
true}
    200880{200880  
If  
Cur-V-WS has a  
non-"ly" suffix in  
BW-Set}
    200882{200882  
If  
V-Mod-Pos-Set[  
TEXT, POS, all  
AV-No's, text item] has a  
final word with an "ly"  
suffix}
    200884{200884  
If  
such a text item  
has its adverbial  
subclass role =  
V-Mod's}
    200886[200886  
Text-Form = 1st  
non-"ly" suffix or 1st  
entry in BW-Set of  
Cur-V-WS;  
Go to 200852;]
    200888{200888  
If  
LY-WS  
is false}
    200890[200890  
LY-WS = True;  
POS-LY = POS;  
LY-Text = 1st  
BW-Set of  
Cur-V-WS;]
    200892{200892  
If  
POS-Vec  
has an untried  
position}
    200876_2[200876;]

    200860 -- T --> 200862
    200860 -- F --> 200870
    200862 --> 200868
    200868 --> 200872
    200872 --> 200874
    200874 -- T --> 200876
    200874 -- F --> 200878
    200876 -- T --> 200872
    200876 -- F --> 200840_2
    200878 --> 200879
    200879 -- T --> 200886
    200879 -- F --> 200880
    200880 -- T --> 200886
    200880 -- F --> 200882
    200882 -- T --> 200884
    200882 -- F --> 200886
    200884 -- T --> 200886
    200884 -- F --> 200888
    200886 --> 200888
    200888 -- T --> 200890
    200888 -- F --> 200878
    200890 --> 200892
    200892 -- T --> 200878
    200892 -- F --> 200876_2

```

FIG. 24U

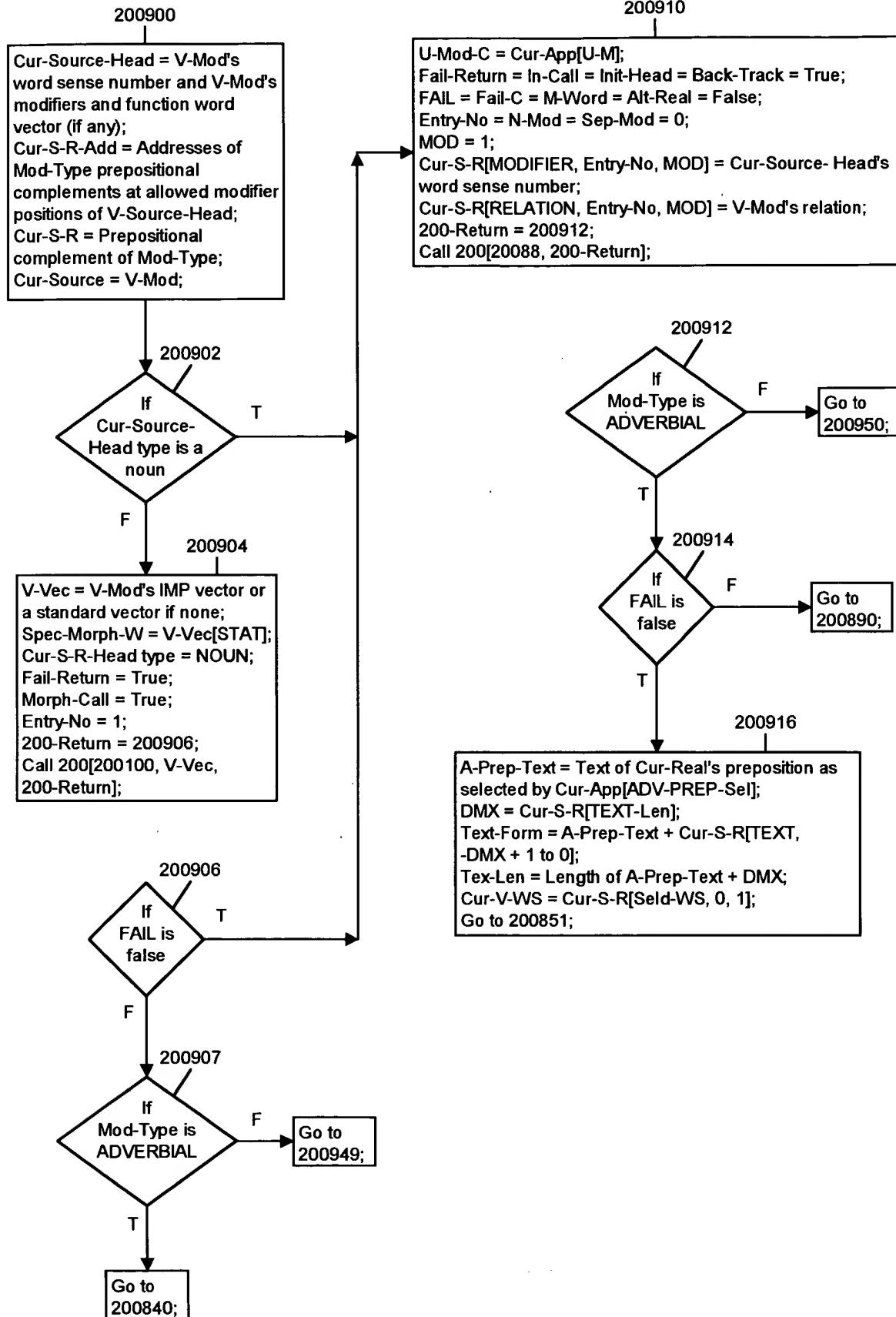


FIG. 24V

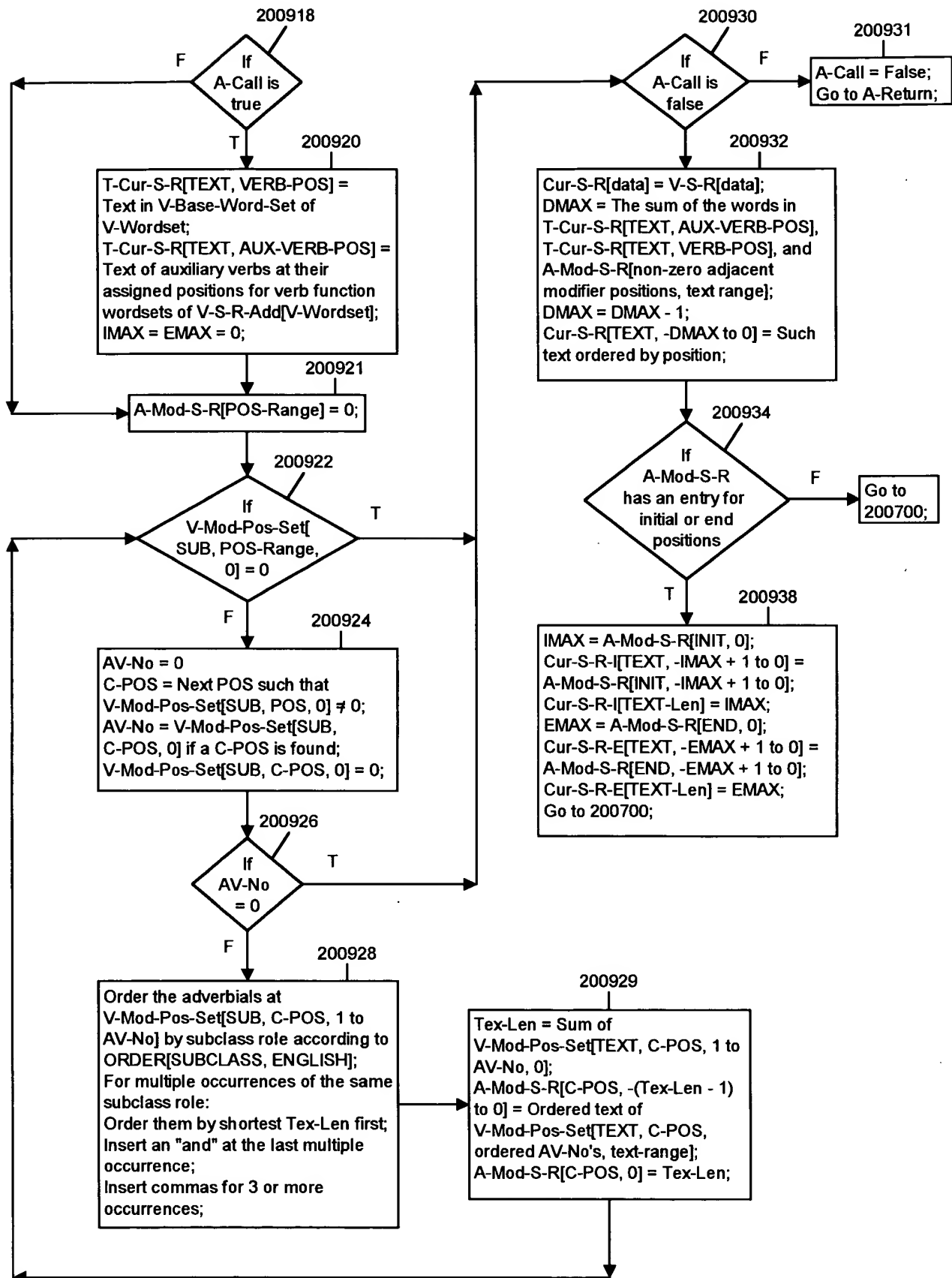


FIG. 24W

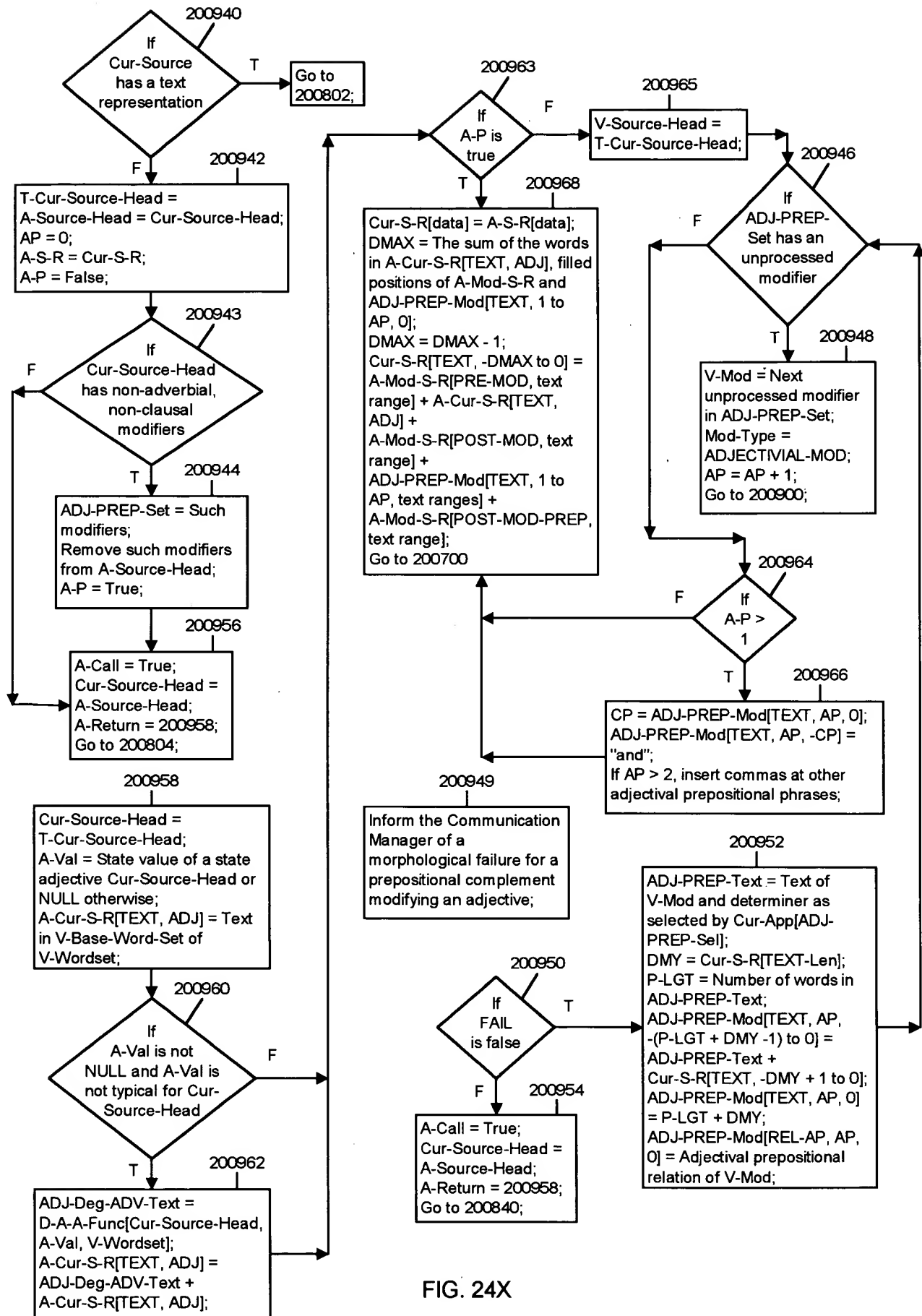


FIG. 24X

```

graph TD
    200700[200700  
Sentence-Check = False;  
Coord-Check = False;  
Mod-Check = True;  
Check-Mod = Cur-S-R[TEXT,  
-DMAX to 0];  
Ellip-Call = True;  
RETURN = 200702;  
Call 200[200200, RETURN];] --> 200702[200702  
DMAX = Remaining words at  
Check-Mod;  
DMAX = DMAX -1;  
Compress Check-Mod to  
Cur-S-R[TEXT, -DMAX to 0];]
    200702 --> 200704{200704  
If  
Next-Out[Cur-S-R,  
Nex-O-Pos] has  
coordinated  
heads}
    200704 -- F --> 200718[Go to  
200718;]
    200704 -- T --> 200706{200706  
If  
Cur-Source  
is the last  
head}
    200706 -- T --> 200710[200710  
DMAX = Words at Cur-S-R;  
DMAX = DMAX -1;  
Compress Cur-S-R to  
Cur-S-R[TEXT, -DMAX to 0];  
CONJN = Text form of the  
conjunction code at  
Next-Out[Cur-S-R, Nex-O-Pos];  
Cur-S-R[TEXT, -DMAX] = CONJN;]
    200706 -- F --> 200712{200712  
If  
Next-Out[Cur-S-R,  
Nex-O-Pos] has  
more than two  
heads}
    200712 -- F --> 200716[200716  
Cur-S-R = Sentence role of next head;  
Cur-Source = Next head at  
Next-Out[Cur-S-R, Nex-O-Pos];  
Cur-Source-Head = The word sense  
numbers of the head and its modifiers  
and its function word vector;  
Cur-Imp-V = Cur-I-V[Cur-Source];  
Go to 20078;]
    200712 -- T --> 200714[200714  
Add a comma to  
Cur-S-R[TEXT, 0];]
    200714 --> 200716
    200716 --> 200718
    200718 --> 200700

```

FIG. 24Y

```

graph TD
    200718{200718  
If  
Cur-Imp-V[CONJ]  
≠ 0 and IC is  
true}
    200720[200720  
Add Cur-Imp-V[CONJ]  
to Cur-S-R[TEXT,  
-DMAX];  
IC = False;]
    200722{200722  
If  
Cur-Imp-V[CONJ]  
is coordinating}
    200724[200724  
COORD = True;]
    200719{200719  
If  
there is an  
unprocessed sentence  
role at Cur-Cla-Add or  
EMAX ≠ 0}
    200721[200721  
Add punctuation of  
Cur-Imp-V[PUNC] to  
Cur-S-R[TEXT, 0];]
    200726[200726  
Cur-S-R[TEXT-Len] = DMAX + 1;  
SDS[Current] = Cur-S-R[TEXT,  
-DMAX to 0];]
    200728{200728  
If  
there is an  
unprocessed sentence  
role at Cur-Cla-Add}
    200730{200730  
If  
there is  
another address at  
SDSO-Pos}
    200731[200731  
A-S-C-Vec[SDSO-Pos] = 0;]
    200732{200732  
If  
IMAX = 0}
    200733[200733  
Cur-S-R[TEXT, -IMAX + 1 to 0] =  
Cur-S-R-I[TEXT, -IMAX + 1 to 0];  
Remove Cur-Imp-V[CONJ] from  
SDS[Current];  
IMAX = 0;  
IC = True;  
Go to 200700;]
    200734{200734  
If  
EMAX = 0}
    200735[200735  
Cur-S-R[TEXT, -EMAX + 1 to 0] =  
Cur-S-R-E[TEXT, -EMAX + 1 to 0];  
EMAX = 0;  
Go to 200700;]
    200736[200736  
Sentence-Check = True;  
Coord-Check = COORD;  
COORD = False;  
Mod-Check = False;  
Ellip-Call = True;  
RETURN = 200738;  
Call 200[200200, RETURN];]
    200738[200738  
Compress text in SDS[Current];  
Adjust text lengths;  
Out-Text = SDS[Current];]
    200740{200740  
If  
A-S-C-Vec  
has a 1}
    200743[200743  
Output OUT-TEXT;]
    200744{200744  
If  
Owning-Pro-V has  
a 0 or Owned-Pro-V  
has a 0}
    200746[200746  
Return  
to caller;]
    200748{200748  
If  
Owning-Pro-V  
has a 0}
    200750[200750  
Cur-Pos =  
Position of 1st  
zero in  
Owning-Pro-V;]
    200752[200752  
Cur-Pos =  
Position of 1st  
zero in  
Owned-Pro-V;]
    200754[200754  
Cur-O-Clause =  
Next-Out[CLAUSE, Cur-Pos];  
Next-S = Next-S + 1;  
S-Cla-No = 0;  
Go to 20002;]

    200718 -- F --> 200719
    200718 -- T --> 200720
    200720 --> 200722
    200722 -- T --> 200724
    200722 -- F --> 200726
    200724 --> 200726
    200719 -- F --> 200721
    200719 -- T --> 200728
    200721 --> 200726
    200726 --> 200738
    200728 -- T --> 200730
    200728 -- F --> 200732
    200730 -- F --> 200731
    200730 -- T --> 200732
    200731 --> 200732
    200732 -- F --> 200733
    200732 -- T --> 200734
    200734 -- F --> 200735
    200734 -- T --> 200736
    200736 --> 200738
    200738 --> 200740
    200740 -- T --> 200746
    200740 -- F --> 200743
    200743 --> 200744
    200744 -- F --> 200746
    200744 -- T --> 200748
    200748 -- F --> 200752
    200748 -- T --> 200750
    200750 --> 200754
    200752 --> 200754
    200754 --> 20002

```

FIG. 24Z